PCT/EP 98/02950

09/463795

Europäisches Patentamt | European Patent Office | Office européen des brevets

5

Bescheinigung | Certificate | Attestation

Die angehefteten Unterlagen stimmen mit der ursprünglich eingereichten Fassung der auf dem nächsten Blatt bezeichneten europäischen Patentanmeldung überein.

The attached documents are exact copies of the European patent application described on the following page, as originally filed.

Les documents fixés à cette attestation sont conformes à la version initialement déposée de la demande de brevet européen spécifiée à la page suivante.

Patentanmeldung Nr. Patent application No. Demande de brevet n°

97112951.5

PRIORITY DOCUMENT

Der Präsident des Europäischen Patentamts:
Im Auftrag

For the President of the European Patent Office

Le Président de l'Office européen des brevets
p.o.

M.W. Graham

D HAAG, N
T HAGUE, 03/06/98
L AYE, LE

EPA/EPO/OEB Form 1014 - 02.91

Europäisches Patentamt
European Patent Office
Office européen des brevets

Blatt 2 der Bescheinigung
Sheet 2 of the certificate
Page 2 de l'attestation

Anmeldung Nr.:
Application no.:
Demande n°: 97112951.5

Anmeldetag:
Date of filing:
Date de dépôt: 28/07/97

Anmelder:
Applicant(s):
Demandeur(s):
IDT INTERNATIONAL DIGITAL TECHNOLOGIES DEUTSCHLAND GmbH
D-85737 Ismaning
GERMANY

Bezeichnung der Erfindung:
Title of the invention:
Titre de l'invention:
Method and apparatus for compressing video sequences

In Anspruch genommene Priorität(en) / Priority(ies) claimed / Priorité(s) revendiquée(s)

Staat: State: Pays:	Tag: Date: Date:	Aktenzeichen: File no. Numéro de dépôt:

Internationale Patentklassifikation:
International Patent classification:
Classification internationale des brevets:
/

Am Anmeldetag benannte Vertragstaaten:
Contracting states designated at date of filing: AT/BE/CH/DE/DK/ES/FI/FR/GB/GR/IE/IT/LI/LU/MC/NL/PT/SE
Etats contractants désignés lors du depôt:

Bemerkungen:
Remarks:
Remarques:

METHOD AND APPARATUS FOR COMPRESSING VIDEO SEQUENCES

Related applications

The application is related to the following application assigned to the same applicant as the present invention and filed on even date herewith, the disclosure of which is hereby incorporated by reference:

Method and apparatus for compression of image residuals. (Our file: IDT 019 EP)

Field of invention

This invention relates to compression of sequences of digital video frames, so as to allow more efficient transmission or storage of the sequences.

Background of the invention

The PCT patent application "Method and Apparatus for data analysis", international publication number WO 95/08240, which is hereby included by reference, discloses a technique, called IDLE, for modeling video sequences, based on bi-linear models of motion and intensity.

The PCT patent application "Method and Apparatus for coordination of motion determination over multiple frames", PCT application number PCT/EP 96/01272, which is hereby included by reference, discloses a technique for coordinating motion estimation with bi-linear modeling.

These two patent applications describe techniques that can be efficient for modeling data, in particular video sequences. However, they do not describe a complete, simple, efficient video encoder based on the IDLE principle.

The patent application "Apparatus and Method for Decoding Video Images", PCT application number PCT/EP 95/02105, PCT international publication number WO 95/34172, which is hereby included by reference, discloses how an IDLE model can be decoded.

Hybrid encoders for video sequence compression, based on Discrete Cosine Transform (DCT) and motion compensation, exist. One example is the MPEG-2 standard. This type of video encoders have a relatively simple structure, but in some applications higher compression ratios are wanted.

Objects of the invention

It is an object of the present invention to provide a technique for compressing sequences of video frames so that a high compression ratio is achieved.

It is a further object of this invention to provide an encoding technique with a relatively simple structure.

It is a further object of this invention to provide mechanisms necessary in a full-featured system, like handling of color, interlacing and random access.

It is a further object of this invention to provide a encoder producing a representation that can be used for interactive purposes.

Notation and definitions

DA (Displacement in Adress): Motion field.
DV and DH (Displacement Vertically and Horizontally): Vertical and horizontal component of motion field.

"Transmit" will here be used as also covering the meaning "store" on some storage medium.

By "fidelity" is meant how well a reconstuction reconstructs an original, expressed in dB, RMS or other easily measurable units. By "quality" is ment the subjective, visual quality of the reconstruction, which in addition to fidelity takes into account how annoying various types of image artifacts are for the human visual system.

Some principles are illustrated using programming code fragments inspired by Matlab, as described in "Matlab Reference Guide", The MathWorks, 1992, which is hereby included by reference.

Summary of the invention

The invention is based on dividing a video sequence into temporal blocks of frames. Each temporal block of frames is modeled according to the IDLE principle by choosing one frame as reference image and representing the other frames in terms of the reference image and a bi-linear model of motion. The reference image and the bi-linear model of motion can themselves be compressed.

Various enhancements to this basic principle are possible: Residuals can be transmitted. The length of the temporal blocks can be made adaptive to sequence content. The rank of the bi-linear model of motion can be made adaptive. Bi-directional modeling can be used to improve compression ratio or image quality. Color can be added. Interlacing can be handled. Changes in intensities can be modeled. Shaped objects can be modeled. Redundancy across temporal blocks can be exploited.

Brief description of the figures

Fig. 1 gives an overview over the invention;
fig. 2 shows the relation between motion estimation and motion modeling;
figs. 3 a - d illustrate the concept of bi-linear modeling;
fig. 4 illustrates a decoder;
fig. 5 illustrates using the first frame in each temporal block as reference frame;
fig. 6 illustrates using the center frame of each temporal block as reference frame;
fig. 7 illustrates the context for updating a bilinear model of motion;
figs. 8 a - d illustrate adaptive determination of temporal block length;
fig. 9 illustrates the use of P frames;
fig. 10 illustrates the context for determining rank of the bilinear model of motion;
figs. 11 a - d illustrate the motivation for using an extended reference image;
figs. 12 a-h illustrate one method for estimating an extended reference image;
figs. 13 a-c illustrate the need for bi-directional modeling;
fig. 14 illustrates the use of B frames;

fig. 15 shows one way of encoding using bi-directional modeling;
fig. 16 shows a decoder adapted to bi-directional modeling;
figs. 17 a - b show two possible contexts for calculation of residuals;
fig. 18 shows how residual dampening may be integrated into the system;
figs. 19 a-d illustrate four ways of handling interlacing;
fig. 20 illustrates the context for bi-linear modeling of intensity;
figs. 21 a - b show how bi-linear modeling of intensity can be implemented;
figs. 22 illustrate preconditioning of intensity changes before bi-linear modelling;
figs. 23 a - d illustrate how subsampled color channels can be decoded;
fig. 24 shows how bi-linear modeling of intensity can be done in color; and
fig. 25 illustrates a decoder for interactive video based on controlling the IDLE model.

First preferred embodiment

ENCODER

Fig. 1 shows the invention in a first preferred embodiment. An encoder Enc 100 receives as input a sequence of video frames 101 and transmits a representation of the sequence as an IDLE model 191. The encoder has a module DivIntoTempBlo 110 for dividing the sequence into temporal blocks of frames. One method is to use a fixed block size, so that each temporal block consists of e.g. nFra=15 frames.

The encoder has a module BuildMod 120 for building an IDLE model of the temporal block of frames.

The IDLE model is transmitted by the module TraMod 130.

The module BuildMod 120 will now be described with reference to fig. 2. BuildMod 200 receives as input the temporal block of frames 201, consisting of a number of frames. One of the frames in the temporal block of frames 201 is selected as a reference image. This can be the first frame. A switch 210 sends this frame into a reference image store RefImgSto 220. The switch 210 sends each of the other frames to a motion estimator EstMov 230. Motion is estimated from the reference image to each of the other frames. This results in nFra-1 motion fields.

The motion estimator EstMov 230 can be based on several principles. One possibility is to use a block based motion estimator, finding, for each block of pixels in the reference image, a motion vector to a similar block of pixels in the frame.

Another possibility is to use a gradient-based method for calculating optical flow, as described in "Determining Optical Flow", B. K. P. Horn and B. Schunk, "Artificial Intelligence" 17, 1981, which is hereby included by reference.

Yet another possibility is to use mesh-based motion fields, as described in "An H.263-compatible system for video coding based on a triangular mesh", Y. Altunbasak, A. Murat Tekalp, International Picture Coding Symposium, 1996, which is hereby included by reference.

The motion fields are sent to a module BuildBLM 240 that builds a bi-linear model of the motion.

The collection of the reference image RefImg 291 and the bi-linear model of motion BLMMotion 292 together consitute an IDLE model 293.

The concept of a bi-linear model of motion, as produced by BuildBLM 240, will now be described with reference to figs. 3 a to d.

Motion fields generally consist of one motion vector for each pixel or each block of pixels in a reference image. In 2-dimensional images, the motion fields have one vertical and one horizontal component. The vertical component of a motion field, DV 310, having size nv vertically and nh horizontally, denoted nvxnh, can be strung out as a row vector 312 of size 1 vertically and nv*nh horizontally.

The row vectors may be collected in a matrix D 350. Using bi-linear modeling techniques, this matrix can be represented as the matrix product of a scores matrix T 352 and a loadings matrix P 354. Optionally, in case the product of the scores matrix T 352 and the loadings matrix P 354 does not reconstruct the data matrix D 350 sufficiently well, a residual E 356 can be added.

Several methods exist for transforming from the matrix D 350 into score and loading matrices. Some methods are described in H. Martens and T. Naes, "Multivariate Calibaration", chapter 3 (John Wiley & Sons, 1989), which is hereby included by reference. In particular, Principal Component Analysis and Karhunen-Loéve transform can be used.

Yet another method is Singular Value Decomposition, in short SVD. SVD represents a data matrix X as the product of three other matrices: $X = U*S*V^T$. T and P can be found by setting either $T=U*S$ and $P=V^T$, or $T=U$ and $P=S*V^T$, or any other way of distributing the singular values S on the left and right eigenvectors U and V^T respectively. Alternatively, the U, S and V matrices may be kept separate.

Scores, loadings, U or V need not be orthogonal. In particular, in case a power method is used for computing the bi-linear model, then the iteration may be stopped before full convergence.

The way of forming each row vector 312 as outlined in fig. 3 a leads to one bi-linear model of motion for the vertical dimension. The same procedure can be used for the horizontal dimension. This is called separate modeling.

Alternatively, as illustrated in fig. 3 b, the vertical and horizontal component of the motion fields DV 330 and DH 332 can be strung out as one row vector 334 jointly. This leads to one bi-linear model of motion describing motion in both the vertical and horizontal dimension. This is called joint modeling.

Alternatively, the motion can be represented as magnitude and angle. One bi-linear model for magnitude and one for angle can be used, or they can be modeled together jointly. Or the angle can be kept constant, and only the magnitude is modeled using a bi-linear model.

The vertical and horizontal component of a motion vector can also be considered as a complex number, and the modelling can be performed using complex SVD or similar methods.

In the following, both one bi-linear model covering of both spatial dimensions of motion using joint modeling, and a collection of one bi-linear model for each spatial dimension of motion, will be

referred to as a bi-linear model of motion. A distinction between the two will be made when necessary.

Further information regarding bi-linear modeling of motion can be found in the patent applications "Method and Apparatus for Data Analysis", international publication number WO 95/08240, and "Method and Apparatus for coordination of motion determination over multiple frames", international publication number WO 96/29679, both already included by reference.

As illustrated in fig. 3 d, one row d_n 370 of the matrix D may be reconstructed by multiplying one row t_n 372 of the scores matrix with the loading matrix P 374, and adding an optional corresponding residual row vector e_n 376. The corresponding motion field can then be reconstructed according to the relationship outlined in figs. 3 a or 3 b.

The collection of one column of the score matrix and one row of the loading matrix is called a factor. The number of factors is called the rank of the model.

The actual storage format used for the scores and loadings may be chosen differently from the above description, which was chosen to give a mathematically based definition of the bi-linear modeling used in this invention. E.g, each loading may be stored as a two-dimensional image. This might be called an eigen-motion.

Usually, the rank can be chosen to be significantly lower than the number of frames in the corresponding temporal block. Also, for many video sequences, the motion residual E 356 will contain small values that are either relatively inexpensive to compress, in terms of bits, or can be dropped without too much degradation in image quality. In total, this gives a potential for compression, in that a low number of score and loading vectors, T 352 and P 354, and optionally an easily compressible motion residual matrix E 356, can be transmitted instead of the full motion fields D 350. The input sequence can then be represented using the reference image and the bi-linear model of motion.

In this preferred embodiment, the motion residual matrix E 356 is not transmitted.

The rank can be set to a constant. E.g, for a temporal block size nFra = 15 frames, the number of factors nFact can be set to 3.

One simple implementation of TraMod is to just transmit the data on a given communication channel.

DECODER

An IDLE decoder suiting to the encoder described above will now be described, with reference to fig. 4. The decoder Dec 400 receives an IDLE model 401 with its ReceiveMod module 410. The reference image part of the IDLE model is sent to the reference image store RefImgSto 420. The bi-linear model of motion is sent to the module RecMov 430, where motion fields are reconstructed from the bi-linear model. The reconstruction of a motion field corresponding to a frame is performed by multiplying scores corresponding to the frame with the loadings. The reference image and the motion fields are then used by a Move module 440 to provide a reconstruction 491 of each of the original frames of the temporal block.

The above procedure is applied for each temporal block.

For the parts of the IDLE model that were compressed on the encoder side, ReceiveMod 410 must decompress accordingly.

The Move module 440 moves the image content of a reference image according to a motion field. This is often called motion compensation. When applied with different motion vectors to individual pixels, it is often called image warping. Several methods for doing this are given in "Digital Image Warping", third edition, George Wolberg, ISBN 0-8186-8944-7, IEEE Computer Society Press, 1994, which is hereby included by reference.

A related method is to interpolate motion between motion vectors computed by block matching, as described in "Feedback Loop for Coder Control in a Block-Based Hybrid Coder with Mesh-Based Motion Compensation", Jörn Ostermann, Proceedings of ICASSP '97, Munich, IEEE Computer Society Press 1997, which is hereby included by reference.

Our Move algorithm

Another method is given in the PCT patent application "Apparatus and Method for Decoding Video Images", international publication number WO 95/34172, pages 31-43, which is hereby included by reference. A full description of an IDLE decoder is given in the same patent application.

Second preferred embodiment

The IDLE model produced by the encoder may be compressed. The overall compression ratio will then be the ratio between the size of the original sequence and the size of the resulting IDLE model, multiplied with the compression ratio obtained for the IDLE model.

The reference image is a still image, and may be compressed using one of many still image compression methods available. In particular, transform coding may be used. One such method, based on the Discrete Cosine Transform, DCT, is JPEG, as described in "JPEG Still Image Compression Standard", William P. Pennebaker, Joan L Mitchell, ISBN 0-442-01272-1, 1993, which is hereby included by reference. Other methods are vector quantization, predictive coding, fractal coding and other still image coding techniques as mentioned in "Digitale Bildcodierung", Jens-Rainer Ohm, Springer Verlag 1995, which is hereby included by reference.

The score and loading vectors can be compressed, as part of TraMod 130 from fig. 1. Various compression methods can be used.

Each loading usually has some character of still images; in particular, there is usually a spatial redundancy. Hence each loading can be compressed using a still image compression tool. Any of the still image compression techniques mentioned above can be used.

Depending on characteristics of other parts of the system, in particular the motion estimator, it can also be advantageous to low-pass filter and subsample some or all of the loadings. Especially the first loadings usually have high spatial redundancy, implying a potential compression or computation gain by subsampling.

The subsampling may be done as part of the compression and decompression so that it is invisible for the other parts of the encoder and decoder.

Alternatively, the loadings may be processed in subsampled format in more modules on the encoder or decoder side. This may give an advantage regarding necessary memory capacity. For example, all the loadings and all the motion fields can be kept in subsampled format all the time in the decoder. The operations on the bi-linear model, as described in fig. 3, remain the same. In this case, the Move operator must be able to handle subsampled motion fields.

"Method and Apparatus for Decoding Video Images", already included by reference, describes a group of Move modules for operating in full resolution for both intensity and motion. One possible modification of these algorithms, suitable for operating in full resolution in intensity and reduced resolution in motion, will now be given by reference to the following pseudocode:

```
(1) For each Group of three or four elements of the motion fields:
(2)   For each Element of Group:
(3)     Compute moved coordinates by adding motion to current position,
(3a)    and multiplying the result by a factor corresponding to
(3b)    the subsampling

(4)   For each Edge of Group:
(5)     Compute DDA parameters for scanning scan lines

(6)   For each ScanLine inside moved coordinates of Group:
(7)     Compute DDA parameters for scanning pixels between edges

(8)     For each Pixel on ScanLine:
(9)       Compute corresponding pixel adress in From-image
(10)      Interpolate an intensity value
```

The above pseudocode is similar for full and reduced resolution of the motion field. One difference is in line 3, where lines (3a) and (3b) are added for the case of reduced resolution. The same factor must be considered when computing the DDA parameters in lines (5) and (7).

The scores for the first factors often have high autocorrelation in time. Hence they can be compressed efficiently by predictive coding in time direction. Another efficient compression method for such data is to use a transform into frequency. Possible frequency transforms include one-dimensional DCT and wavelet packages.

In case the scaling between scores and loadings is made so that the energy of each factor is included in the scores, e.g. by normalizing each loading, the size of the scores should be considered when compressing the loadings. E.g, a loading that will only be multiplied with small scores in a decoder will need less fidelity than a loading that will be multiplied with large scores.

One method is to scale the scores so that the score vector for each factor has the same value for some norm, e.g. the maximum value. The loadings are scaled accordingly, so that the product of scores and loadings stays the same. If the loadings are now transmitted with the same fidelity, then the maximum impact of the quantization error will be the same for all factors.

Several still image compression tools have fixed input ranges. E.g. if the bi-linear modelling is made so as to normalize the loadings to unity, while the still image compression tool can only handle

integer input values in the range 0-255, then the loadings will have to be shifted and scaled. The fidelity of the compression tool should be adjusted correspondingly. E.g, for a system using still image compression based on DCT, then the quantization of the DCT coefficients should be adjusted according to scaling and scores.

Third preferred embodiment

In this embodiment, the reference image can be selected adaptively.

In the first preferred embodiment, the first frame of each temporal block was selected as a reference image, as basis for motion estimation.

The resulting structure is illustrated on fig. 5. One reference frame, denoted I 510, is used as a reference image. The other frames, denoted U 520 530 540 550, are represented as motion compensated, also called moved, versions of the reference image. The motion fields 560 are modeled using bi-linear modeling as described above.

Other frames can be chosen as reference images. Fig. 6 shows a center frame 630 of a temporal block of frames, with both earlier 610 620 and later 640 650 frames represented as moved versions of the center frames. The motion fields 660 can be modelled together. Alternatively, one bi-linear model can be used for the earlier frames 610 620 and one model can be used for the later frames 640 650.

Another possibility is to search for the most representative frame in the temporal block and use this as a reference image. One way to find a representative frame is to compute an average or median histogram over intensity or color for all the frames, and then selecting the frame whose histogram has the highest similarity with an average or median histogram.

Selecting a suitable frame to be used as reference image for a temporal block can also be done using analysis by synthesis: Several frames can be tried as reference images, and the selection that results in the most efficient modelling can be accepted.

Fourth preferred embodiment

In this embodiment, the bilinear model of motion is updated recursively.

In the first preferred embodiment, the motion fields were first estimated for each frame of the temporal block. Afterwards, the bi-linear modeling was performed.

In some applications, memory consumption may be important, and then it would be advantageous to have a method that does not need intermediate storage for all motion fields. Or it may be important to have parallel algorithms, so that motion estimation and calculation of bi-linear models can occur more or less simultaneously. For such applications, the bi-linear models can be updated for each frame instead of being built at the end of a temporal block.

This is illustrated on fig. 7. The BuildMod module 700 again takes a temporal block of frames 701 as input, a switch 710 sends one frame to a reference image store RefImgSto 720 and the other frames to a motion estimator module EstMov 730. The resulting motion fields from EstMov 730 are now sent to a module for updating a bilinear model, UpdateBLM 740. At the start of a temporal

block, a model store BLMSto 750 for a bi-linear model is initialized to zero. At each new motion field, UpdateBLM 740 reads the current bi-linear model from BLMSto 750, updates the bi-linear model, and stores it back.

Several methods for updating a bi-linear model exist. Some are given in "Multivariate Calibration", H. Martens & T. Næss, John Wiley & Sons, 1989, which is hereby included by reference.

Fifth preferred embodiment

In this embodiment of the invention, the length of each of the temporal blocks is selected adaptively.

Referring back to fig. 1, the encoder Enc 100 had a module DivIntoTempBlo 110 for dividing the input sequence 101 into temporal blocks. This module will now be described with reference to fig. 8 a - d.

For related or similar motion fields, bi-linear modeling can bring advantages, e.g. with regard to compression ratios. However, when motion fields with little similarity are modeled together, the bi-linear modeling may bring disadvantages, both regarding fidelity vs. compressability for compression of motion and for fidelity vs. compressability for the total system. It may therefore be advantageous to adapt the temporal blocks in time.

Motion fields often have little similarity across scene shifts. One way of implementing DivIntoTempBlo is therefore to use a scene shift detector. Several methods for detecting scene shifts exist. One is to detect differences between histograms of pixel values for two or more consecutive frames. Another is to detect scene shifts manually.

Another is to perform a motion estimation between two consecutive frames, and define a scene shift when the mean or RMS Displaced Frame Difference exceeds a given threshold. This is described in "Scene Context Dependent Reference Frame Placement for MPEG Video Coding", A. Lan, J. Hwang, ICASSP '97, IEEE Computer Society Press, 1997, which is hereby included by reference.

When a scene shift, indicating the start of a new temporal block, has been found, BuildMod 140 passes its current IDLE model, relating to the previous temporal block, on to TraMod, so that it can be transmitted. In the case of an updated bi-linear model, as illustrated in fig. 7, the bi-linear model in BLMSto *UpdMod*50 must be re-initialized to empty.

Fig. 8 a shows how the motion estimation module EstMov 730 from fig. 7 or 230 from fig. 2 can be used for defining temporal blocks, in addition to its normal function. Similarily to fig. 7, BuildMod 800 receives a temporal block 801 of frames as input, one frame is directed by a switch 810 into a reference image store RefImgSto 812 and the others are given to EstMov 814. Now the motion fields from EstMov 814 are used to move the reference image in the Move module 816. The absolute value of the difference between the reference image and the frame is summed up in SumAbsDiff 818. For related images, this will often give a relatively small result, while for unrelated images, this will often give a large result. The sum is fed to the encoder control EncCont 822, where it can be compared with a given threshold. As long as the sum stays below the threshold, the temporal block is prolonged; in particular the motion fields are input to BuildBLM 820 so that they are included in a bi-linear model of motion. When the sum is above this threshold, this defines the end of the temporal block. The reference image 821 and the bi-linear model of motion 822, together constituting an IDLE model 823, are given as output.

The similarity between an original image and a reconstruction of the image using a model can also be expressed in terms of spatial size of model failure areas. One possibility is to calculate an intensity residual image as the difference between the original image and the reconstruction, divide the residual image into spatial blocks, and then count the number of spatial blocks where the average error of the spatial block is above a given threshold. The average can be found by sum of absolute values of the residual image divided by number of pixels in the spatial block, or by RMS, or by another norm. Other characterizations can also be used, like a median value. A threshold can be used to distinguish between model success and model failure spatial blocks. Another threshold can then be used for the number of allowed model failure spatial blocks, e.g. so that the temporal block is prolonged as long as the number of model failure spatial blocks stays below 5 % of the total number of blocks in the frame.

Another way of finding the spatial size of model failure areas is to calculate the intensity residual image, and apply a threshold to this image, and count the number of pixels which are above the threshold. The thresholded image will often correspond more to a human judgement if the residual image is slightly blurred before thresholded, or the thresholded image is processed with morphological filters like Open and Close.

Fig. 8 a illustrates the use of forward motion compensation for determining temporal block limits. A similar effect can be achieved using backward motion compensation, also called moving back, as shown in fig. 8 b, where a module BuildMod for building an IDLE model is illustrated.

A temporal block of frames 831 is input, one selected frame is directed by the switch 832 so that it is stored in the reference image store RefImgSto 834, where it is the basis for motion estimation in EstMov 835. Now, the current frame is moved back in moduled MoveBack 836 using the motion field estimated by EstMov 835. The result is expected to be similar to the reference image for the case of highly related images. Therefore the result from a module 838 for summing absolute differences between the frame and the reference image can be compared against a given threshold in the encoder control EncCont 839, again possibly trigging a new temporal block.

When defining model failure spatial blocks, characteristics of the human visual system may be considered, so that prolongation of temporal blocks is based on an estimate of how visible artifacts will be, instead of the reconstruction fidelity. Methods from "Method and apparatus for compression of Image Residuals", already included by reference, may be used to discriminate between highly and less visible artifacts in the intensity residual image. Residuals corresponding to artifacts below a visual threshold may then be removed from the residual image before each block is characterized as model success or model failure according to the above procedure.

This is illustrated on fig. 8 c. A module BuildMod 850 for building a model is shown. A temporal block 851 of frames is given as input. The function of the switch 860, EncCont 869, RefImgSto 861, EstMov 862, MoveBack 863 and BuildBLM 864 is as described for fig. 8 b.

In addition, there is a module DefBounds 866 for defining bounds for how strong image artifacts must be in order to be visible. This module can exploit masking effects of the human visual systems. Examples include entropy masking and nonlinear response for intensity. It produces an upper and a lower bound for each pixel or block or pixels. A fuller description is given in "Method and Apparatus for compression of image residuals", already included by reference.

There is also a module RelToBounds 867. It relates the difference between the original and reconstructed frame to the bounds. In the figure, each frame is moved back to reference position by MoveBack 863 and then the difference to the reference image is made in the difference module 865, producing a residual. This is the basis for relating to bounds. The differencing can also be made in frame position, in which case the bounds can be moved to the frame position according to the estimated motion.

RelToBounds 867 can be based on several principles. One is to check whether the residual is within the bounds, producing a yes/no result, e.g. encoded as 0 if the residual is within the bounds and 1 otherwise. Another is to compute the ratio between the residual and the upper or lower bound. Optionally, if the ratio is smaller than 1, then it can be set to zero. Yet another is to compute the difference between the residual and the upper or lower bound.

A summation module 868 then sums the results of RelToBounds 867, giving the results to EncCont 869, which can take decisions regarding length of temporal blocks based on these results.

Alternatively, the fit of a new motion field to a bi-linear model can be used for deciding whether to start a new temporal block. This is illustrated on fig. 8 d, showing a module BuildMod 875. The results from EstMov 884 are projected on a bi-linear model of motion in ProjOnBLM 886, producing scores and a residual. If the sum of absolute values of the residual, as computed in SumABS 888, is above a threshold, then EncCont 874 may define a new temporal block of frames. Otherwise, UpdBLM 892 updates the bi-linear model of motion stored in BLMSto 890. The projection produces a set of scores and a motion residual. Thresholding on sum of squares of the residual can be used for defining a new temporal block: When the sum of squares is small, the motion field fits well into the bi-linear model.

The threshold can be absolute, e.g. corresponding to an allowed average or maximum deviation, expressed in number of pixels, between the motion field as estimated by EstMov 884 and the same motion field as modeled by BuildBLM.

The threshold can also be relative, e.g. corresponding to the proportion of explained vs. total variance for the motion field: If more than a given threshold of the variance of a motion field can be modeled by projecting the motion field on a bi-linear model of motion, then the temporal block can be prolonged.

Absolute and relative thresholding can be combined, e.g. by checking both that the motion deviation introduced by modeling is less than a limit AND that a certain amount of the variance of the motion field can be modeled.

Alternatively, the motion field can be projected on a bi-linear model or a bi-linear model can be updated with the motion field, the result can be used to reconstruct the corresponding frame, and then the check on goodness can be done by comparing the reconstructed frame with the corresponding original frame.

Some methods for updating a bi-linear model have a projection on new data on old factors as one of the first steps. ProjOnBLM 886 can therefore be seen as a first step of UpdateBLM 892.

In some applications, random access into the video sequence may be wanted. Examples include channel change in connection with broadcasting, or fast forward for a storage based system. In such systems, some maximum distance between reference frames can be prescribed. An encoder control

module EncCont can then count the frames, and force reference frames at given intervals. This can be combined with other methods for defining temporal blocks. E.g, a system could have a frame counter. Each time a new temporal block is triggered using any of the methods outlined in figs. 8 a-d, the frame counter is reset to zero. For each frame, the frame counter is increased by one. If the frame counter reaches a given threshold, a new temporal block is triggered and the frame counter is reset to zero. E.g, for a TV system using 25 frames per second and where other components than IDLE modelling contribute with a delay corresponding to at most 5 frames, the threshold may be set to 20 if random access with at most one second delay is prescribed.

In some applications, manual definition of temporal blocks may be wanted. One example is a storage based system where fast forward or backward functions take sequence content into consideration, e.g. so that each push on a fast forward button advances to the start of the next scene in the sequence. Another example is interactive video or games, where user action or reaction influences the selection of frames to be replayed, or even manipulated. In such applications, some access points may be defined by a human operator. Each such access point then forces the start of a new temporal block.

Such defined access points may be combined with other mechanisms for defining temporal blocks. E.g, the system may proceed through the sequence, defining a new temporal block whenever either an access point is prescribed, a scene shift is detected, or a frame counter reaches a maximum value.

Sixth preferred embodiment

The concepts of adaptive temporal block length and adaptive choice of reference frame can be combined.

One method is to first determine the temporal block length according to any of the described methods, and then choose reference frame adaptively.

Another method is based on selecting temporal block length and choosing reference frame simultaneously. For many sequences, it is so that if a first frame can be efficiently reconstructed from a second frame and a motion field, then the second frame can be efficiently reconstructed from the first frame and the inverse motion field. Additionally, if there are frames between these two frames, and each can be efficiently reconstructed from the first frame, then they can often also be reconstructed from the second frame.

These assumptions form the basis for the method illustrated in the following pseudo-code:

```
(1)   Select first frame of the temporal block as preliminary reference
      image PrelRefImg

(2)   Set the current frame CurrFra to the first frame
(3)   Repeat
(4)     Set CurrFra to the next frame
(5)     Estimate motion DA from PrelRefImg to CurrFra
(6)     Move PrelRefImg according to DA, giving CurrFraRec
(7)   until CurrFraRec not similar to CurrFra

(8)   Set RefImg to the frame before CurrFra
```

```
(9)  Set CurrFra to RefImg
(10) Repeat
(11)   Set CurrFra to the next frame
(12)   Estimate motion DE from RefImg to CurrFra
(13)   Move RefImg according to DA, giving CurrFraRec
(14) until CurrFraRec not similar to CurrFra

(15) Define the frame before CurrFra as the last frame of the
     temporal block
```

The aim is a representation as illustrated in fig. 6, such that the temporal block will consist of as many frames as possible. Lines (1) - (8) will find a frame RefImg 30, to be used as reference image, that is as far away from the first frame 10 while still maintaining the possibility to reconstruct one from the other. Lines (9) - (15) will then find the end of the temporal block, such that the all frames after RefImg can still be reconstructed from RefImg.

Additionally or alternatively, the tests in lines (7) and (14) could be based on features of the motion field, so that the motion field stays smooth or fulfills other criteria for being "reasonable".

One such criterium could be that the motion field fits well into a bi-linear model of motion, with a limited number of factors, that is updated for each frame.

Alternatively, the motion DA used for moving in steps (6) could be the part of the motion estimated in steps (5) that can be reconstructed from a bi-linear model of motion, with a limited number of factors, that is updated for each frame.

Seventh preferred embodiment

There may be cases where a sequence is broken into multiple temporal blocks, though some similarity between the frames remain between the temporal blocks. One such similarity is when the reference image of a first temporal block is similar to the reference image of a second temporal block. The second reference frame can then be reconstructed as a moved version of the first reference image, plus an intensity residual.

The structure is illustrated in fig. 9. A first reference frame 910 is used as basis for motion estimation to several frames 912 914 916 918. The motion fields 940 can be modeled together as a bi-linear model. One frame 918 is then used as a reference image for motion estimation to further frames 920 922 924 926 928. One 928 of these can once more be used as a reference for motion estimation to further frames 930 932 934.

A related technique, but without the bi-linear model of motion, is used in the MPEG class of encoders.

One method is to first encode each temporal block independently. Then, as part of model compression, the reference image of one temporal block can be encoded as a motion compensated version of another. In this case the encoder transmits one motion field plus a residual, expressing one reference image in terms of the other. Forwards or backwards motion may be used.

Another method is to let the motion from the first to the second reference frame be included in the bi-linear model of motion for the temporal block corresponding to the first reference frame. The following pseudo-code describes the operation:

```
IRefCurr=Move(IRefPrev,ScoPrev(CurrIndex,:)*LodPrev)+IResCurr
```

where IRefPrev is the previous reference image, ScoPrev is the scores of the previous model, CurrIndex is the temporal index of the new reference image in the previous model, LodPrev is the loads for of the previous model, IResCurr is a residual related to this process, and IRefCurr is the resulting new reference image for the current model.

The current reference image is covered by the previous temporal block. Therefore the motion field from the previous reference image IRefPrev to the current reference image IRefCurr can be found from the bi-linear model of motion for the previous temporal block by multiplying the scores corresponding to the current reference image, ScoPrev(CurrIndex,:), with the loads, LodPrev. The previous reference image IRefPrev is then motion compensated according to this motion field. A residual, IResCurr, is added, and the result is the current reference frame IRefCurr.

Eighth preferred embodiment

Different temporal blocks may have a different optimal number of motion factors nFact, also called rank of model. E.g, when the frames of a temporal block only show a still image, without motion, then zero motion factors are needed. On the other hand, when a long temporal block contain complex, but highly systematic, motion, like a human head talking, higher compression ratios may be obtained by building a bi-linear model that captures much of the motion, although needing several factors.

In principle, motion information is lost in several stages in this system. First, motion estimators will in general not be able to estimate the true motion. Second, by using less than full rank in the bi-linear modeling, some of the motion will normally not be represented in the bi-linear model. Third, when the bi-linear model is compressed, more motion information is lost unless a lossless compression technique is used. This preferred embodiment illustrates aspects of making a good compromise between compression ratios and fidelity or quality for the second stage of loss, namely rank reduction.

One way of adapting the rank, is to first update or build a bi-linear model using a higher number of factors, and then determining a final rank after all frames of the temporal block have been processed. This is illustrated in fig. 10. BuildMod 1000, corresponding to the module 140 of same name in fig. 1, has a module UpdateBLM 1050 for updating a bi-linear model stored in BLMSto 1060. Before the bi-linear model is passed on from BuildMod 1000, the rank of the model is determined in DetRank 1070.

One simple assumption is that longer temporal blocks may have more different motion phenomena than short temporal blocks. DetRank may then be based on a set of rules or a table giving the number of factors to be used depending on the temporal block length. Such a set of rules could be:

```
if nFra <= 1
  nFact = 0
elseif nFra <= 4
  nFact = 1
elseif nFra <= 10
  nFact = 2
else
  nFact = 3
```

Alternatively, DetRank 1070 may be based on the content of the bi-linear model. One method is to define a minimum energy for a factor, and discard those factors that have an energy below a threshold. The energy of a factor is given e.g. as the corresponding singular value.

Another method is to define a smallest relevant motion. E.g, one may define that motions smaller than one quarter pixel are not visible, and therefore cut away those factors whose largest absolute value of the score multiplied with the largest absolute value of the loading do not exceed 0.5.

Another method is to define a relative energy threshold. E.g, if the energy of a small factor has less than 5 % of the energy of the largest factor, then the small factor is discarded.

Alternatively, the mechanisms listed for DetRank 1070 might be performed as part of UpdateBLM 1050. This may have advantages in computation time or storage requirements, in that UpdateBLM 1050 and BLMSto 1060 only need to consider the selected number of factors.

The image content may have an influence on the necessary fidelity of the motion model. E.g, in textureless areas of the image some "false" motion may be of low visual importance, while in textured parts of the image even small errors in motion may be visible.

One way of adapting to image content is to decode the IDLE model of the temporal block using different number of factors. The number of factors to transmit is then chosen as a compromise between image fidelity or quality achieved with some number of factors, achievable improvements in image fidelity or quality by transmitting an additional factor, and number of bits required for transmitting the additional factor.

Considering image content for determining rank can alternatively be done by characterizing allowable deviations of the motion for different parts of the image. One method is to calculate intensity gradients in the reference image. E.g, if the horizontal intensity gradient is close to zero, then a small deviation in horizontal motion will be insignificant with regard to reconstruction quality, while a similar small deviation in horizontal motion for an image area with a large horizontal intensity gradient might result in a highly visible artifact.

The intensity gradients can be used as a basis for weights in building a bi-linear model of motion: Areas with large intensity gradients are given larger weights than areas with small intensity gradients.

Alternatively, the intensity gradients can be used for setting a pixel-wise tolerance for the fidelity of the representation of motion as a bi-linear model.

The intensity gradients may be computed using various methods, including differences between neighbouring pixels, a set of Sobel filters, a Kirsch operator or another of the methods described in "The Image Processing Handbook", J. Russ, CRC press, 1995, page 233-249, which is hereby included by reference.

Instead of operating on intensity gradients for judging allowable motion deviations, the motion slack can be counted. Motion slack at a given slack level in a given direction is defined as how much motion can deviate from a given point estimate in that direction, without the corresponding resulting intensity exceeding the slack level.

Determining the rank can alternatively be regarded as a part of the compression of the bi-linear model of motion. During compression of a bi-linear model of motion, each loading can be coded separately using an image-compression tool. This tool can be controlled so that only significant parts of the loading are transmitted. E.g, if the image-compression tool is based on dividing the loading into spatial blocks and transform coding each block using DCT, then it can be checked if the motion related to this factor and block is below a given threshold, say, one quarter pixel. If it is below, then the block is skipped, otherwise it is transform coded. The number of factors to be used can then be defined by the last factor whose loading had a significant number of non-skipped spatial blocks.

Alternatively, the rank of the model may be determined by considering the singular values, or the sum of squares of loadings or scores. Often, the first factors have mainly to do with "signal", and the last factors have mainly to do with "noise". The "noise" in this case includes motion estimation errors, quantization errors, compression artifacts and effects from thermal noise in the image. Often, the factors mainly related to "signal" will have strongly decreasing singular values, while the factors related to "noise" will have a much flatter spectrum. The transition from strongly decreasing to flatter distribution can often be seen as a "knee point" on a plot of singular value as a function of factor number. An encoder can choose to discard factors with a singular value smaller than the singular value of the factor at the knee point.

The methods for determining intermediate rank in UpdateBLM 1050 and the final determination in DetRank 1070 may be combined.

The rank may be reduced from full to intermediate in UpdateBLM 1050, mostly in order to save computation resources, while the rank may be further reduced in DetRank 1070, mostly to optimize compression ratios.

When recursive updating of a bi-linear model is used, then relative small but highly systematic motion may have similar characteristics as noise for the first few frames where the motion is observed, and only when several frames have been observed can the motion be found as a bi-linear factor. Such motion will not necessarily be reflected in the bi-linear model of motion if all noise-

like factors are removed after each update. This phenomenon is one rationale for using a higher rank during the updating of the bi-linear model than the rank that will be used finally.

All of the above methods for determining the rank apply both to the case of joint modeling, with one bi-linear model of motion covering both vertical and horizontal dimension, and separate modeling, where there is one bi-linear model for vertical dimension and another for the horizontal.

Furthermore, for separate modeling the rank can be determined in common for both dimensions, or the rank can be determined independently for each dimension.

Nineth preferred embodiment

In general, it is not assured that all parts of all frames can be reconstructed from one reference frame plus motion data; some frames may have parts that are not covered by moving the reference frame. This is illustrated in figs. 11 a - d. Fig. 11 a shows a reference frame. Fig. 11 b shows a reconstructed frame as a reference frame moved with an arbitrary motion field. There are disappearing parts 1140 1141 1142 1143 of the reference frame. These are in general of little imortance. However, there are also parts 1130 1131 1132 1133 of the frame to be reconstructed that are not covered by the moved reference frame.

One way to cope with uncovered parts is to transmit a residual for those parts. One can choose to handle the uncovered parts like other model failure regions. This has an advantage with regard to algorithmic simplicity.

Another way is to create an extended reference image, containing the reference frame plus new content. This is illustrated in fig. 11 c. The same reference frame as was shown in fig. 11 a has now been added more pixels 1160 1161 1162 1163, thus forming an extended reference image, so that the reference image covers the whole frame to be reconstructed, as illustrated in fig. 11 d. The bi-linear model of motion must be adjusted correspondingly.

One method for creating an extended reference image is based on extrapolating the loadings of the bi-linear model of motion, and extending the reference image according to pixels from the frame moved back according to the extrapolated loadings. This method will now be described in more detail, with reference to figs. 12 a-h. An original reference image IRefOrg 1200 is given, depicting a first tree 1202 and a person 1204. A frame In 1210 is also given, depicting the same person 1212 but a second tree 1214, due to camera panning. A motion field 1220 is calculated from IRefOrg 1200 to In 1210. One 1222 of the motion vectors for the tree and one 1224 of the motion vectors for the person in the reference image have been drawn. The task is now to prepare a new reference image with place both for the person 1204 1212, the first tree 1202 and the second tree 1214.

A padded version 1230 of IRefOrg is created, so that it contains IRefOrg 1200, with additional room around it so that pixels from In 1210 can be filled in. One conservative method is to find the minimal and maximal value of the vertical component of the motion field 1220, MinDV and MaxDV respectively, and minimal and maximal value of the horizontal component of the motion

field 1220, MinDH and MaxDH respectively. If MinDV < 0, then IRefOrg is padded with abs(MinDV) pixels on the upper border, and if MaxDV > 0, then IRefOrg is padded with MaxDV pixels on the lower border. Similar padding is done in the horizontal direction. The result is named IRefPad 1230.

The same padding is performed on the motion field 1220, producing a padded motion field 1240.

The padding may imply a change of coordinate system. To correct for this, the motion vectors can be compensated. E.g, if the origin of IRefOrg was in the upper left pixel and the origin of the padded image shall also be in the upper left pixel, then values corresponding to the padding should be subtracted from the motion field: If IRefOrg was padded with abs(MinDV) pixels on the upper border, then abs(MinDV) is subtracted from each motion vector. The same is done in the horizontal direction.

Next, the motion loadings in the padded regions is extrapolated from the given motion fields. The extrapolation should be done so as to minimize discontinuities, both between padded and original areas and inside padded areas. One useful technique is the following diffusion process, performed for each loading:

```
(1)  For each pass:
(2)    SNew = SOld
(3)    INew = IOld
(4)    For each v:
(5)      For each h:
(6)        If SOld(v,h) == 0:
(7)          Sum = 0
(8)          SumWgt = 0
(9)          For each dv in (-1, 0, 1):
(10)            For each dh in (-1,0,1):
(11)              If dv!=0 and dh!=0:
(12)                If SOld(v+dv,h+dh) <> 0
(13)                  SumWgt=SumWgt+1
(14)                  Sum=Sum+LodOld(v+dv,h+dh)
(15)         If n>0:
(16)           LodNew(v,h) = Sum/SumWgt;
(17)           SNew(v,h) = 1
(18)   LodOld = LodNew
(19)   SOld = SNew
```

The algorithm starts with a "processed" area containing valid motion, defined by a given S field. For each pass trough the field, all pixels that are adjacent to the processed area are set to an average of those neighbouring pixels that are inside the processed area.

A weighting can be used, e.g. replacing lines (13) and (14) by

```
(13)                  SumWgt = SumWgt + WgtTable(dv,dh)
(14)                  Sum=Sum+WgtTable(dv,dh)*LodOld(v+dv,h+dh)
```

where WgtTable is a table giving the nearness to the central pixel v,h, e.g. using values like

```
      dh

    2 3 2
dv  3   3
    2 3 2
```

Now an extended reference image 1250 can be formed by moving the content of In back according to the motion field, for those areas of IRefPad where no pixels is stored. In pseudo-code:

```
(1) For each scan line v in IRefPad:
(2)   For each pixel on the scan line h:
(3)     If S(v,h) == 0:
(4)       If Sn(v+DV(v,h), h+DH(v,h)):
(5)         IRefOrg(v,h) = In(v+DV(v,h), h+DH(v,h))
(6)         S(v,h) = 1
```

If the motion field DA=DV,DH is given with subpixel accuracy, then this can be exploited in line (5). One method is to calculate the subpixel position (v+DV(v,h), h+DH(v,h)), and set IRefOrg(v,h) to a weighted sum of the four surrounding integer position pixels, where the weights are based on nearness to the four surrounding integer positions. This and other methods are described in "Digital Image Warping", George Wolberg, IEEE Computer Society, third edition, 1994, which is hereby included by reference.

The padding that was not necessary can now be removed, giving the final results extended reference image 1260 and the final corresponding motion loadings 1270. This can be done by finding the smalles rectangle containing the S field produced according to the above pseudocode.

The S fields may be compressed by any technique for compressing bi-level images. Some possibilities are constant area coding, white block skipping, one- and two-dimensional run length coding, contour tracing and predictive differential quantizing, as described in "Digital Image Processing", R. Gonzales and R. Woods, Addison-Wesley Publishing Company, 1992, pages 351-362, which is hereby included by reference. Other possibilities are chain coding, differential chain coding and polygonal approximations as described on pages 484-488 of the same book, which is hereby included by reference. Another possibility is JBIG, as described in "A simple and efficient binary shape coding technique based on bitmap representation", F. Bossen and T. Ebrahimi, Proceedings of ICASSP '97, IEEE Computer Society Press, 1997, which is hereby included by reference.

The S field may be utilized when compressing the other spatial parts of the IDLE model. E.g. if the reference image is compressed using a spatial block-based compression system based on e.g. DCT and the S field has already been transmitted to the decoder, then the encoder and decoder can trivially skip all the spatial blocks that are completely outside the S field, with zero overhead in transmitted bits.

The existence of an S field can also be exploited for partially filled spatial blocks. E.g, partially filled spatial blocks of the reference image or each loading may be compressed using any of the techniques described in "A comparison of efficient methods for the coding of arbitrarily shaped image segments", Proceedings of the International Picture Coding Symposium 1996, I. Donescu, O. Avaro, C. Roux, page 181-186, which is hereby included by reference.

Tenth preferred embodiment

There are many sequences that cannot be modeled well with only one reference frame, or even an extended reference image. Consider figs. 13 a - c. A first frame 1300 in a temporal block shows a woman 1305 partly occluded by a car 1310, together with a man 1315. A last frame 1360 shows the woman 1365 fully, but this time the man 1375 is partly occluded by the car 1370. An intermediate frame 1330 shows the woman 1335 and the man 1345 partly occluded by the car 1340. For this very short sequence it is not possible to select one reference frame that can be used to reconstruct both the woman 1305 1335 1365 and the man 1315 1345 1375 fully in both sequences.

Assuming that the two frames 1300, 1350 were the first and last frames in a long temporal block, it would probably be possible to reconstruct each part of the intermediate frame using either the first 1300 or last 1360 frame.

This leads to the principle of bi-directional modeling. One early (e.g. the first) and one late (e.g. the last) frame of a temporal block are both selected as reference frames. The temporal block is modeled once using the early frame as reference frame, leading to a forward IDLE model, and once using the late frame as reference frame, leading to a backward IDLE model. For each spatial block of pixels in each of the intermediate frames, it is measured whether the forward or backward IDLE model produces the best reconstruction. The encoder sends one bit for each spatial block in each intermediate frame, telling the decoder whether to use the forward or backward IDLE model for that spatial block in the intermediate frame.

The structure is illustrated in fig. 14. A reference image 1412 is used as basis for motion estimation and compensation to nearby frames 1410 1411 1413 1414 1415 1416. Another reference image 1416 also covers nearby frames 1414 1415 1417 1418 1419 1420 1421. Those frames 1414 1415 that are covered by two models are called bi-directional frames, or B frames in short. Those frames 1410 1411 that are only covered by one model are called unidirectional frames, or U frames in short. The structure can be continued, in that a new reference image 1421 covers more frames 1417 1418 1419 1420 1422 1423 1424.

If an I frame 1425 is inserted, e.g. because of random access reasons, B frames 1422 1423 1424 may still be used. However, if a decoder then became access to the bit stream after the transmission of the previous I frame 1412, then the B frames 1422 1423 1424 will have to be decoded as if they were U frames to be decoded only from the new I frame 1425. A similar possibility exists for error concealment: If a part of a reference image was lost or destroyed during communication or storage, then the corresponding part of a reconstructed B frame might be taken from the other reference image only.

A system for encoding into this structure is illustrated in fig. 15. An encoder Enc 1500 receives as input a sequence of frames. The sequence is divided into temporal blocks by DivIntoTempBlo 1510. A module BuildModForw 1520 builds a forwards IDLE model of each temporal block, using an early frame as reference frame. A module BuildModBack 1530 builds a backwards IDLE model of the same temporal block, using a late frame as reference frame. The frames of the temporal block are reconstructed 1540 1550 using the forwards and backwards IDLE model respectively, and the quality of each reconstruction is assessed 1560 1570. For each pixel or each spatial block of pixels of each frame, a B field value is calculated in CalcBField 1580, such that the B field has one value, e.g. 0, for each pixel or each spatial block of pixels that is to be taken from the forwards model and another value, e.g. 1, from each pixel or each spatial block of pixels that is to be taken from the backward model. The forward IDLE model, the backward IDLE model and the B fields, together comprising a full IDLE model, are then passed on to TraMod 1590.

The B fields will often have spatial redundancy: If one block of the reconstruction is taken from the forward IDLE model, then there is an increased probability that the neighbouring block will also be taken from the forward IDLE model. Each B field can therefore be compressed using a compression tool for bi-level images. Any of the methods that were given for S fields in the nineth preferred embodiment can also be used for B fields.

The B fields will also often have temporal redundancy: If one block in one frame was taken from the backward IDLE model, then it is an increased probability that the corresponding block of the next frame will be taken from the backward IDLE model. This can be exploited e.g. using also elements from the previous B field as part of the template in a JBIG-like method.

The temporal and spatial redundancy of the B fields occur for areas of the frames where the modeling is good in one direction and less good in the other. In areas of the frames where both the forward and backward models produce close to equally good quality or fidelity, then the selection of the best will start being influenced by minor effects, like thermal noise in the images. The switching back and forth between forward and backward models might then introduce visual artifacts in the reconstructions, and also the B fields will be less compressible. It can then be advantageous to apply a technique that will simplify the B fields in those areas where the forward and backward reconstruction are of similar quality.

One such technique is to determine in which areas the forward and backward models have similar fidelity or quality, and then median filter the B fields in those areas.

Another technique is to adapt the B value for each pixel according to a compromise between lack of fit and similarity to neighbours. One method, called Agree, for doing this is given in the following pseudocode:

```
(1)  For each Pixel:
(2)    B(Pixel) = The value that minimizes lack of fit
(3)  For each subsequent pass:
(4)    For each scan line:
(5)      For each Pixel on the scan line:
(6)        BNew(Pixel) = the value that minimizes a sum of lack
(7)        of fit and dissimilarity with neighbours
(8)    B = Bnew
```

Lines (1) and (2) make the optimal choice for each block of pixels independently. However, both regarding compressability of the B fields and regarding visual artifacts, particularily on the borders of the B fields, it may be preferable to sacrifice some of the fit for better coherency of the B fields. Lines (3) to (7) therefore attempts to make the B fields more coherent by making a compromise between fit and similarity with neighbours. This is done by looping over pixels, in line (3) and (4), and for each pixel selecting the B value that minimizes the sum of lack of fit and dissimilarity with neighbours. One example of such an expression is:

$$\min_{B=0,1} \left(\sum_{Pel} (I[B](Pel) - I(Pel)) + k * \sum_{Neigh} (B \text{ xor } B(Neigh)) \right)$$

What is minimized is a weighted sum of two quantities. The first quantity is a sum of lack of fit, summed over all pixels within the block of pixel. The second quantity counts how many of the neighbouring blocks have a B value that is different from the one sought.

To speed up convergence and to save memory consumption, the iteration in line (4) and (5) can be made over every second scan line and every second pixel, which allows line (6) to be made using replacement instead of copying, which again makes line (8) superfluous:

```
(1)  For each pixel:
(2)    B(pixel) = The value that minimizes lack of fit

(3)  For each subsequent pass:
(4)    For every second scan line, starting on alternate
       scan lines:
(5)      For every second pixel on the scan line, starting on
         alternate pixels:
(6)        B(pixel) = the value that minimizes a sum of lack
(7)        of fit and dissimilarity with neighbours
```

The above technique, which was shown for block-wise decisions on whether to use the forward or backward model in the reconstructed frames, can also be used with block size 1, that is, for each pixel. Then the conditioning of the B fields, e.g. as described for the Agree filter, and compression of the same fields, become more important.

Another technique to reduce visual artifacts from the transition between forwards and backwards model is to use overlapping blocks for the B field. The B field can be seen as a weight field. Instead of using only either 0 or 1 weights, the weights can be feathered. Consider a transition

from backward to forward model in the middle of eight consecutive pixels on a scan line. Without feathering, the weights for the forwards model may be:

0 0 0 1 1 1

while the weights for the backwards model are:

1 1 1 0 0 0

With feathering, this may become, for the forwards model:

0 0 0.25 0.75 1 1

and for the backwards model:

1 1 0.75 0.25 1 1

The B fields can also be extended to allow other weights than 0 and 1 to be transmitted. E.g, also allowing a reconstructed block to be formed as an average between the reconstructions from the forward and backward model respectively, an encoder could for each block of the frame to be reconstructed send one code for average, one code for forward or one code for backward. Also in this case Agree, median filtering or similar techniques can be used for improving the compressability of the B fields.

A decoder must decode both the forward and backward IDLE model, and for each block decide whether to use the forwards or backwards reconstruction. This is illustrated in fig. 16, showing a bi-directional decoder Dec 1600, receiving a bit stream 1601 containing a forward model, a backward model, and B fields. A switch 1610 directs the forward model to a model store 1620 and frame reconstruction module 1630 for decoding in the forwards direction and the backward model to a model store 1640 and frame reconstruction module 1650 for decoding in the backwards direction. For each block of each decoded frame, a Select module 1660 selects the reconstruction from either the forward or backward model.

There must not necessarily be two basic decoders operating in paralell. One decoder can be multiplexed. E.g, there could be one basic decoder, first decoding the frames from the forward model and storing them, and then decoding the frames from the backward model. Blocks can so be selected using the B fields.

The same applies for the encoder. There may be two basic encoders, or there may be one multiplexed between forward and backward direction.

Eleventh preferred embodiment

There may be cases where a frame cannot be well reconstructed as a motion compensated reference image. Some special cases could be solved, like occlusions and innovations, as described in the previous preferred embodiment, or objects that enter or leave the scene during a temporal block, as described using extended reference images. But many cases, including cases of

unsystematic motion or intensity changes, still cannot be modeled efficiently. These areas may have visible modeling artifacts, which in many applications are not tolerated. Such areas are called model failure areas.

Model failure areas can be corrected by transmitting a residual. One method is to simulate the decoding in the encoder, producing reconstructed frames. The difference between the reconstructed frames and the original frames can then be transmitted.

This is illustrated on fig. 17 a. An encoder Enc 1700 is shown. A sequence 1701 of frames is given as input. DivIntoTempBlo 1705 divides them into temporal blocks. BuildMod 10 builds an IDLE model of the frames. The quantized model is so transmitted by TraMod 1725. The quantized model is also used for reconstructing the frames in a module Rec 1715. A differencing module 1720 calculates the difference between the original and the reconstructed frames, producing a residual. This is then transmitted in TraRes 1730.

Alternatively, quantization or other compression artifacts coming from lossy compression of the model can be considered when computing the residuals. This is illustrated on fig. 17 b, showing an encoder Enc 1750, receiving frames 1751, dividing 1755 them into temporal blocks and building 1760 an IDLE model. Quantization, thresholding or other types of lossy data reduction normally taking place as part of data compression and transmission is so performed in module CompMod 1765. The resulting model is used to reconstruct frames in the module Rec 1770. The difference 1775 is calculated. TraMod 1780 and TraRes 1785 transmit the model and the residual respectively.

Different parts of an image may have different tolerances regarding how large residuals are allowed at a given quality level, due to human visual system masking effects. Some such masking effects, and also ways of exploiting such effects when transmitting residuals, are described in the patent application "Method and apparatus for compressing image residuals", already included by reference.

To achieve the highest possible fidelity, the residuals should be computed on the encoder side based on a full simulation of the decoder. That is, the encoder should incorporate all artifacts of compression and decompression of the reference images and the bi-linear model of motion. Only then is it guaranteed that transmitting the residuals will indeed improve the fidelity of the reconstructed frames.

When the reference image is compressed with a lossy technique, it will have artifacts. These artifacts will then manifest themselves also in the motion compensated versions of the reference image, which implies that one compression artifact may have to be corrected in residuals for several frames. This will degrace the compression ratios for the residuals.

Artifacts in the reference image will result in systematic errors in a temporal block, since the motion is modeled by a common bi-linear model for the entire temporal block. In many cases, these systematic errors will be of visually little importance, which makes them acceptable in some applications. Therefore the residual can be computed as if the reference image had been losslessly compressed. That is, for purposes of computing residuals, the original reference image is used instead of that coming from CompMod 1765.

A similar argument can be used for motion. Slight artifacts in motion might not be easily visually detectable, but still lead to residuals when the computation of residuals include a full simulation of the compression and decompression. Therefore, the residual might be computed as if the loadings had been transmitted losslessly.

A similar argument also applies regarding the rank of the bi-linear model of motion. When the encoder chooses a number of factors, this is done as a compromise between bit rate and quality. In some situations, some factors that are not transmitted describe motion that is visually insignificant, but whose removal still leads to an increase in residuals. In these situations, it may be beneficial to compute the residuals as if a fuller rank had been used for the bi-linear model of motion.

Characteristics of the current motion field and the bi-linear model of motion may be considered when estimating how visible artifacts in motion will be. E.g, a spatially and temporally smooth error in motion, with a magnitude of e.g. 1 pixels, may be hardly visible in a larger area with a smooth motion field, whereas a motion artifact with a magnitude of 0.5 pixels might be easily visible in an area with a non-smooth motion field. Such information can be exploited when deciding whether original or quantized motion should be used.

Twelfth preferred embodiment

The P frames may be transmitted as motion scores relative to an earlier reference image and corresponding bi-linear model of motion, plus a residual image. This residual image can in principle be treated like a B-frame as in the previous embodiment, in that visually insignificant residuals may be suppressed or dampened. Principles according to "Method and apparatus for compression of image residuals", already included by reference, may therefore be used for dampening the residuals, so that only visually significant residuals are transmitted.

The visibility of image artifacts depends on how long the artifacts remain, among other things. In case the method used for dampening artifacts is tuned to one maximal duration, then artifacts that remain longer may not be caught by the mechanism, but still remain visible. One method for overcoming this is to toggle between dampening residuals and not dampening residuals for P-frames.

This is illustrated on fig. 18, showing an encoder Enc 1800. A sequence 1801 of frames is given as input. DivIntoTempBlo 10 divides the frames into temporal blocks. BuildMod 1820 builds an IDLE model of the frames. CompMod 1830 compresses the model, which is then transmitted in TraMod 1880. Rec 1840 reconstructs the frames from the quantized model. The difference module 1850 calculates the difference between the original frame and its reconstruction, producing a residual signal. A switch 1860, controlled by the encoder control module EncCont 1805 is toggeled between sending the residual through a module DampRes 1870 for dampening the residuals, or sending the residuals directly to Trans 1890 where the residuals are transmitted.

The toggeling can be done so that residuals are transmitted undampened for a certain fraction of the P frames, e.g. every second P frame.

Alternatively, the toggeling can be done with respect to time, expressed in number of frames or in seconds. E.g, the toggeling can be done so that if more than one second has elapsed since the last I or P frame was transmitted without dampening of residuals, then dampening should be disabled for the next P frame, else it should be enabled.

Alternatively, the encoder can control the parameters to DampRes instead of operating a switch. For the case of the method given in "Method and apparatus for compression of image residuals", already included by reference, a bit rate control parameter affecting bounds for allowed residuals can be used. The parameters can be set so that P frames are normally transmitted with strong dampening of residuals, while some chosen P frames are transmitted without or with very weak dampening of residuals.

This may be extended to several levels, e.g. so that P frames are normally transmitted with strong dampening of residuals, some are transmitted with weaker dampening of residuals, and some are transmitted without dampening of residuals.

Thirteenth preferred embodiment

For many applications, the input sequence will be given in interlaced format, where each frame consists of two fields, one even and one odd.

The system as described in previous embodiments can be used directly also on interlaced sequences, though suboptimal bit rates or visual artifacts may result. This may have advantages regarding simplicity.

Alternatively, the system can be made to operate on fields instead of frames. This means that the reference image is chosen as one field, and the other fields are modeled as motion compensated versions of this reference image. This is illustrated on fig. 19 a. The frames 1910 1915 are split into fields 1920 1921 1922 1923, and this sequence of fields is modeled as if they were a sequence of frames, e.g. using the first field 1920 as reference frame and estimating motion to the other fields 1921 1922 1923, and then building a bi-linear model of the motion fields.

This may have advantages regarding simplicity.

Alternatively, the system can operate with one model for even fields and one model for odd fields. This is illustrated on fig. 19 b. Frames 1925 1930 are split into even 1935 1936 and odd 1937 1938 fields. The IDLE method is then applied once to the even fields 1935 1937 and once to the odd fields 1936 1938.

This may have advantages regarding parallelizability, though the bit rate at a given quality may not be as good as for all other methods for all sequences.

Alternatively, instead of building two separate models, one for even and one for odd fields, the motion corresponding to the two fields of one frame can be modeled jointly. Referring to fig. 3 c, this would mean that each row in the D 350 matrix would consist of one motion field for the even field of the reference image and one motion field for the odd field of the reference image,

concatenated after each other. Each loading then has elements both for the even and for the odd field. Referring to fig. 3, this implies that when when scores t_n 372 for a frame are multiplied with the loadings P 374, and optionally a residual e_n 376 is added, the result d_n 370 is a vector containing one motion field suitable for reconstructing the even field of the frame from the even field of the reference image, and one motion field suitable for reconstructing an odd field of the frame from the odd field of the reference image.

Alternatively, a reference image of the same spatial resolution as each frame may be formed. This is illustrated on fig. 19 c. A frame 1950, containing an even 1951 and an odd 1952 field, is split into two images 1960 1961, each based on one field. The missing scan lines in each image 1960 1961 are filled using interpolation between the available scan lines. The same is done to a next frame 1955 also containing and even 1956 and an odd 1957 field, resulting in two new image 1962 1963. IDLE modelling is so performed on the new images 1960 1961 1962 1963.

Since scan lines have been interpolated, they will in general not correspond perfectly with what would have been registered if the camera had been non-interlaced. If intensity change modeling is used, like will be described for the next preferred embodiment, the intensity change model may contribute to compensating for the lost resolution. Further, if a score for a intensity change factor remains close to constant over time, this is an indication that the factor may be compensating for the lost resolution, and this factor may then be added permanently to the reference image instead of being transmitted as a separate factor.

Prior to or as part of IDLE modeling, each frame can be deinterlaced. This is illustrated on fig. 19 d. A first frame 1975, containing even 1976 and odd 1977 fields, is split into two new images 1981 1982 by copying the given scan lines and interpolating the rest. Motion 1985 is then estimated between the two images 1981 1982. One high-resolution image 1988 can then be formed from the two images, e.g. by keeping the even scan lines from the image 1981 based on the even field and moving the content of the image 1982 based on odd lines back to fill in the missing scan lines. The same procedure is performed for the next frame 1978, giving one image 1983 based on even scan lines and one 1984 based on odd scan lines, combined into one high-resolution image 1989. The high-resolution images 1988 1989 can be modeled using IDLE.

The motion field 1986 between the even and odd field of the second frame can be moved back to reference position using a motion field 1990 between the two frames. In this case, all motion fields are given in reference position, and will in many cases fit well into a bi-linear model, giving a compression gain.

Fourteenth preferred embodiment

In addition to motion, there may be systematic changes in intensity in the input sequence. In many cases, these intensity changes may be modeled efficiently using bi-linear models. The basic method is explained in the patent application "Method and Apparatus for data analysis", already included by reference.

Bi-linear modeling of intensity can be incorporated in the present invention, as shown on fig. 20. An encoder Enc 2000 takes as input a sequence 2001 of frames. The frames are divided into temporal blocks by DivIntoTempBlo 2030. A bi-linear model of motion is build in BuildModMov 2040. The bi-linear model of motion, together with the reference image and the original frames, form the basis for building a bi-linear model of intensity changes in BuildModInt 2050. The total output is transmitted in TraMod 2060.

The bi-linear model of intensity changes can be combined with the bi-linear model of motion in two ways.

The first is to model intensity changes in reference position, according to the following formula:

$$IRec = Move(IRef + ScoBlu*LodBlu, ScoSmi*LodSmi)$$

To the reference image IRef is added the product of blush scores ScoBlu and blush loads LodBlu. The result is so moved according to the product of smile scores ScoSmi and smile loads LodSmi, producing the reconstruction IRec.

The second is to model intensity changes in frame position, according to this formula:

```
IRec = Move(IRef, ScoSmi*LodSmi) + ScoBlu*LodBlu
```

IRef is first moved according to the product of smile scores and smile loadings. The product of blush scores and blush loadings is so added, producing the reconstruction IRef.

Modeling intensity changes in reference position can have advantages regarding compression performance when moving objects change intensity systematically. Modeling intensity changes in frame position can have advantages regarding implementation simplicity.

The internal working of the BuildModInt will now be described with reference to fig. 21 a, illustrating a mechanism for modeling intensity changes in reference position. A frame In 2101 and an IDLE model consisting of a motion model BLMMov 2102 and a reference image IRef 2103 is given as input. RecMov 2110 reconstructs each motion field from the bi-linear model BLMMov 2102. In 2101 is moved back to reference position using the motion field in MoveBack 2110. The difference between this image and the reference image is calculated 2120. One such difference image for each frame in the temporal block is the input to BuildBLM 2130.

Another way of implementing BuildModInt is shown on fig. 21 b. The same input as in fig. 21 a is given. The reference image IRef is moved 2160 according to the motion model. The difference between the moved reference image and the current frame is computed 2170. This difference is moved back 2180 to reference position, again according to the motion model. One such moved difference for each frame in the temporal block is the input to BuildBLM 2190.

Some types of residuals are visually more disturbing than others. This may be considered when modeling intensity changes. E.g, if a sharp intensity edge is positioned one quarter of a pixel out of position in a reconstruction, this will lead to a large intensity residual, though the artifact may

not be visible. Further, it is a difficult problem to estimate the motion fields for moving objects at the boundary of objects, and the residuals may even be unsystematic in magnitude. Since such residuals are large, they may influence the bi-linear modeling. Since they are unsystematic, they may lead to a high number of factors necessary to obtain good fidelity. Since they occur at edges, they may sometimes be visually insignificant. It is therefore advantageous to let such pixels have less influence on the bi-linear modeling of intensity changes.

One method is to preprocess the intensity residual image before modeling. This is illustrated on fig. 22, showing an alternative BuildModBlu 2200. A reference image IRef 2203, a bi-linear model of motion BLMMov 2202 and each frame In 2201 are given. Motion fields are reconstructed in RecMov 2210, the frame is moved back to reference position in MoveBack 2220, and the difference 2230 between this image and the reference image is fed to Damp 2240 where the residual is dampened, considering masking effects based on the reference image. The dampened difference is then basis for building a bi-linear model in BuildBLM 2250.

Another method is to use weights in the bi-linear modeling. Instead of removing intensity differences of relatively little visual importance from the residual before bi-linear modelling, the intensity differences can be allowed to remain, and the weights in the bi-linear modelling are changed instead. Residuals of little visual importance can then be given low weights. For this purpose, BuildBLM 2250 must be able to accept pixelwise weights for each frame.

The modeling of intensity is dependent on the motion. Three possibilities are given.

A first possibility is to use as input each motion field as calculated by EstMov.

A second possibility is to reconstruct the motion field from the bi-linear model of motion, possibly after determining the rank of the bi-linear model. This was shown e.g. in fig. 21 a.

A third possibility is to reconstruct the motion field from the bi-linear model of motion, after the bi-linear model has been compressed and decompressed.

The three possibilities have an increasing ability to produce a model that will lead to high fidelity in the final decoded image, in that the two last possibilities compensate for the effects of bi-linear modeling and the third possibility further compensates for the effects of compression and decompression of the bi-linear model.

This increased fidelity comes at the expense of increased bit rate, though, since the motion artifacts introduced by bi-linear modeling and compression and decompression may lead to intensity change patterns that not always fit well into a bi-linear model.

Fifteenth preferred embodiment

The principle behind this invention can also be used for for compressing color sequences.

An introduction to color imaging is given in "The Image Processing Handbook", Second Edition, John Russ, IEEE Press, pp. 32-47, which is hereby included by reference.

One compromise between complexity and efficiency is to process and store the reference image in color, in some colorspace like YUV, YIQ, YCrCb, HSV, RGB or similar, while motion estimation is executed on intensity only, e.g. the Y component in YUV or YIQ or YCrCb, or V in HSV. This motion is so applied to all color channels.

The chromacity of the colors in the reference image may be subsampled, similar to what is done in MPEG-2.

On the decoder side, the chromacity may then be upsampled to have the same resolution as the intensity component, and these images are then the input to the Move operator.

This is illustrated on fig. 23 a, showing a decoder Dec 2300. A reference image, represented by its three color components Y 2301 U 2302 and V 2303 is given as input and stored in a reference image store RefImgSto 2310. The U 2302 and V 2303 components are subsampled. A bi-linear model of motion 2305 is given as input. Motion fields are reconstructed from this model in RecMov 2311. The U and V components are upsampled in two Expand modules 2312 2313, and are given together with the full resolution Y component and the motion field to a Move module 2314. The resulting color image represented as Y 2320, U 2321 and V 2322 is output.

In this way, the reference image is stored with subsampled chromacity, and the chromacity is upsampled prior to Move 14. In a systolically operated system, the Expand modules 2312 2313 may operate synchronously with the transfer rate for the motion field and the Y component of the reference image, implying that very little memory is needed for the Expand 2312 2313 modules.

The order may be changed, in that the U and V components may be upsampled prior to being stored in the reference image store RefImgSto. This is illustrated on fig. 23 b, showing a decoder Dec 2325. An intensity signal Y 2326 in full resolution and two subsampled chromacity signals U 2327 and V 2328 are given as input. U 2327 and V 2328 are upsampled and filtered in two Expand modules 2340 2341, before being stored in a reference image store RefImgSto 2342. A bi-linear model of motion 2329 is given as input to a module RecMov 2343 that reconstructs motion, and the motion and reference image are given as input to Move 2344 that produces the final full resolution signals Y 2345, U 2346 and V 2347.

Alternatively, the Move operation can be performed in low resolution for the chromacity components, and the results can be upsampled to match the resolution of the moved intensity image. This is illustrated on fig. 23 c, showing a decoder Dec 2350. An intensity signal Y 2351 in full resolution and two subsampled chromacity signals U 2352 and V 2353 are given as input, and all are stored in a reference image store RefImgSto 2360. A subsampled bi-linear model of motion 2354 given as input, and correspondingly the motion fields produced by RecMotion 2361 are still in reduced resolution. They are therefore upsampled in an Expand module 2362 to fit with the full resolution Y component in the Move module 2364, producing a final full resolution Y signal 2370. The low resolution motion fields fit with the low resolution U and V reference images, and they can therefore be given to a low resolution Move module 2363. The resulting U and V signals must be upsampled in Expand modules 2365 2366, producing the final full resolution U 2371 and V 2372 signals.

Alternatively, a Move module may have Expand capabilities built in. This can apply for the motion field, for the intensities, or for both. One context for a Move taking low resolution images and low resolution motion fiels is shown in fig. 23 d, showing a decoder Dec 2375. A full resolution intensity signal Y 2376 and low resolution chromacity signals U 2377 and V 2378 are stored in a reference image store RefImgSto 2385. A low resolution bi-linear model of motion 2379 is given as input to RecMotion 2386, which reconstruction motion. The motion is given together with U and V to MoveExp 2389, performing at the same time motion and upsampling, producing final full resolution U 2396 and V 2397 signals. The motion is upsampled in an Expand 2387 module to fit with the full resolution Y signal in a Move 2388 module, producing the final full resolution Y 2395 signal.

In case residuals are added to the frames after modeling, the residuals may be given in color. The chromacity can be compressed harder than the intensity, e.g. by reducing the spatial resolution, increasing quantization step sizes etc.

Methods exist for estimating motion based on color images. One method is to use block matching, and instead of minimizing for each block e.g. sum of absolute differences between the intensities of the two images, the sum could be made also over the color channels. This can lead to a win in compression ratio.

Another method is to perform motion estimation in the first principal component of the color reference image, as described in US Patent 5387937, which is hereby included by reference.

In case bi-linear modeling of intensity changes is used, this may be done on each color channel. This is illustrated on fig. 24 showing a module BuildModBlu for building a bi-linear model of intensity changes based on a given bi-linear model of motion. A color frame In, represented in a given color space, e.g. YUV, is given as input InY 2401 InU 2402 InV 2403. A bi-linear model of motion 2404 is given as input to RecMov 2410, reconstruction motion fields. A reference image, represented as three color channels IRefY 2405, IRefU 2406 and IRefV 2407, is moved in a Move 2420 module. For each channel, the difference between the color frame and the moved reference image is computed in differencing modules 2430 2432 2434. The result is moved back to reference position in MoveBack 2440, and one bi-linear model of intensity for each color channel is built in BuildBLM 2450 2452 2454, producing final bi-linear models BLMY 2491, BLMU 2492 and BLMV 2493.

In some methods for moving images, a significant part of the work has to do with calculating pixel addresses, and since much of these calculations will be the same for the color channels, one color-capable Move module may be preferred over three single-color-channel Move module. Such a color-capable Move module is described in "Apparatus and Method for Decoding Video Images", already included by reference.

All methods described above for determining rank of an intensity model for intensity also apply to color. In particular, each color channel can be seen as one single-component channel, and the rank can be determined independently for each color channel.

Alternatively, the rank can be determined in common for the color channels.

Instead of building one separate bi-linear model per color channel, one joint bi-linear model covering all color channels can be built. This may have an advantage in a situation where the intensity factors should correspond well to a real-world-like behaviour, e.g. in games or interactive video. One the other hand, compression performance may not be optimal.

Referring to fig. 3, one common bi-linear model of intensity may be build by letting each row in a data matrix D 350 contain all the color components of each pixel in the image. Correspondingly, the loadings 354 will have one value for each color component for each pixel, and each value of the scores 352 will influence all color channels.

Sixteenth preferred embodiment

There are applications where a constant fidelity or quality is wanted. One example would be a system for replay of video sequences from a Compact Disc, CD. When various sequences demand various bit rates to achieve a wanted quality, this can be compensated by changing the speed of readout from the CD.

A close to constant fidelity can be achieved e.g. by setting a threshold for the transmission of residuals, without compensating for effects of the human visual system, to a wanted value.

A close to constant quality can be achieved by the use of DampRes, as explained earlier.

In other applications, a bit rate is prescribed, and the highest achievable fidelity or quality within this limitation is wanted. The bit rate can be prescribed as a maximum for each frame, for an average over some specified number of frames, or for a complete sequence like a film.

Often there will be a transmission buffer, so that an an encoder can for a short time produce more than or less than the wanted average number of bits per second. On the average though, the number of produced bits per second must be equal to or lower than a given limit. This opens for a feedback solution, where the buffer fullness is used as a basis for controlling parameters in the system.

One parameter that can be controlled to achieve a given bit rate is the fidelity of the residuals, as described earlier.

The fidelity of the IDLE model can be adjusted. If a compression system based on DCT is used, then the quantization of the DCT transform coefficients may be adjusted depending on buffer fullness.

The number of factors, both for motion and intensity, can be changed to adjust bit rate.

In a case where residuals are transmitted, care must be taken so that an intended lowering of bit rate through a lowering of number of factors does not lead to increased residuals, and thereby an increase in bit rate. This can be achieved by computing residuals using all selected computed factors, while still only transmitting a limited number of factors. Thus, some parts of the image will end up being moved less correctly, without this being compensated by intensity or residuals.

The bit rate can in some cases be adjusted by changing the average number of frames in each temporal block. By increasing the average number of frames in each temporal block, possibly while keeping the number of factors constant or only slowly increasing, the bit rate can be decreased, though at a cost in fidelity.

All of these mechanisms can be built into a buffer control system. One possibility is to use feedback based on buffer fullness. In this case, the encoder maintains either a communications buffer or a simulation of such a communications buffer, monitoring how many bits are produced by the encoder and how many bits are consumed by the decoder or transmission system. After finishing compression of a frame or a temporal block of frames, key values describing the buffer are used to control encoding parameters that influence the quality or fidelity versus bit rate.

A general description of control strategies is given in "Automatiseringsteknikk", Arne Tyssø, NKI-forlaget, 1985, ISBN 82-562-1696-4, which is hereby included by reference.

The buffer control can be based on an integral controller: The encoding parameters can be set based on the fullness of the buffer. If the buffer is close to empty at a given time, then the encoding parameters for the following frames are set so as to allow high fidelity or quality. If the buffer is close to full, the encoding parameters are set so as to maintain a low bit rate for the following frames, possibly at the expense of fidelity or quality.

Somewhat similar buffer control systems are used in MPEG-based systems. One proposal, based on adapting both quantizer step size and spacing between I or P frames, is given in "Controlling the Rate of MPEG Video by Dynamic Variation of Sequence Structure", I. Richardson, M. Riley, International Picture Coding Symposium, Melbourne, Australia, 1996, which is hereby included by reference.

This scheme can be combined with a proportional controller: If the number of bits resulting from the previous frame or temporal block of frames was high, this can be used to influence the encoding parameters for the next frames so as to reduce the number of bits to be produced.

The scheme can also be combined with a derivative controller: If the number of bits resulting from the last few frames has been strongly increasing, this can be used to influence the encoding parameters for the next frames so as to reduce the number of bits to be produced.

The scheme can also be combined with a second-order integral controller: If the average fullness of the buffer has been high for a given integration period, then this can also be used to influence the encoding parameters for the next frames so as to reduce the number of bits to be produced.

There may be more than one control loop. E.g, on short term, the transmission of the residuals may be controlled, while on longer term, the quantization of the IDLE model may be controlled.

The manipulated value may influence more than one single encoding control parameter. In particular, it may control a linear combination of encoding control parameters.

Seventeenth preferred embodiment

In some cases, consecutive IDLE models relating to consecutive temporal blocks may have features in common. One way of exploiting this was shown in the seventh preferred embodiment. A similar effect can be exploited for loadings of the bi-linear model of motion:

```
LodCurr = LodPrev + ResLod
```

The loadings of a current temporal block can be seen as a sum of the loadings of a previous temporal block plus residuals, allowing the residual to be sent instead of the full loadings. This may bring a compression gain if the loadings of the previous and current temporal block are similar.

In general, there will normally be motion between the reference images. This can be exploited for the loadings. E.g, consider a face saying something at one position in one temporal block and saying something similar at another position in another temporal block. The bi-linear models of motion will then have factors corresponding to what is being said, but the motion vectors relating to talking will will be located in different spatial parts of the loadings. In such cases, it may be beneficial to compensate for the difference in position by representing the loadings of the current temporal block as a moved version of the loadings of the previous blocks:

```
LodCurr = Move(LodPrev, DA) + ResLod
```

In general, though the subspaces spanned by two temporal blocks may be very similar, the loadings may still be rather different between the temporal blocks. To solve this, a rotation matrix can be used, so that the subspace of the previous bi-linear model is rotated so as to coincide as well as possible with the subspace of the current bi-linear model.

From a decoding perspective, this rotation can be made before applying the Move operator:

```
LodCurr = Move(RotMat*LodPrev, DA) + ResLod
```

In this case, the loadings from the previous block LodPrev are rotated using a rotation matrix RotMat. The result is moved according to a given motion field DA. To this result, a residual ResLod can optionally be added.

In case LodPrev and LodCurr have been computed for two temporal blocks independently on the encoder side, LodPrev has already been transmitted to the decoder, and a motion field DA from the previous reference coordinate system to the current coordinate system is available on both the encoder and decoder side, then RotMat and ResLod can later be computed as follows:

The purpose is to provide a RotMat and a corresponding ResLod so that LodCurr can be reconstructed on the decoder side. For compression purposes, RotMat and LodCurr should be as compressible as possible. Often, RotMat will be a much smaller amount of data than LodCurr, so LodCurr will dominate the amount of data that need to be transmitted. RotMat should then be computed so as to maximize the compressability of ResLod without sacrificing too much fidelity in the reconstruction of LodCurr. One method that performs reasonably well for maximizing the compressability of ResLod is to compute RotMat so that the sum of squares of ResLod is minimized. This can be achieved through the following pseudo-code:

```
(1) For each factor:
(2)   LodCurrAtPrev(factor) = MoveBack(LodCurr(factor), DA)
(3) RotMat = LodCurrAtPrev * pinv(LodPrev)
```

"pinv" is a function for computing a pseudo-inverse of a matrix. One example of pinv is given in Matlab Reference Manual, already included by reference. The implementation of pinv in Matlab is more efficient when the vertical size of the matrix is larger than the horizontal. Consequently, line (3) could in Matlab be implemented as

```
(3) RotMat = pinv(LodPrev')' * LodCurrAtPrev';
```

Not all pixels of the loads of the previous and current reference coordinate systems may be valid. One reasone could be mechanisms as described in the nineth preferred embodiment. Another reason could be that not all pixels of the current reference coordinate system could be reachable by the previous reference coordinate system. In that case, the performance of the above method can be improved by considering S fields. In particular, the operation in line (3) can be made so that only pixels of the previous coordinate system that are valid according both to an S field given for the previous coordinate system and to an S field moved back from the current coordinate system are considered. Thus, the above pseudo code can be extended to:

```
(1) SCurrAtPrev = MoveBack(SCurr, DA)
(2) For each factor:
(3)   LodCurrAtPrev(factor) = MoveBack(LodCurr(factor), DA)
(4) SComm = SCurrAtPrev & SPrev
(5) RotMat = LodCurrAtPrev(SComm) * pinv(LodPrev(SComm))
```

Line (1) moves the S field of the current reference coordinate system back to the previous coordinate system, using the given motion field, producing SCurrAtPrev. Lines (2) and (3) move each loading of the bi-linear model of motion for the current coordinate system back to be previous coordinate system, producing LodCurrAtPrev. Line (4) selects the union of the pixels given by SCurrAtPrev and SPrev, producing SComm. Line (5) now operates only on those pixels of LodCurrAtPrev and LodPrev that are specified in SComm.

Alternatively, the rotation can be made after applying the Move operator. The following expression may form a basis for the operation on the decoder side:

```
LodCurr = RotMat*Move(LodPrev, DA) + ResLod
```

This implies that the factor rotation is performed in the coordinate system of the current reference image. An encoding method suited for this is:

```
(1) SPrevAtCurr = Move(SPrev, DA)
(2) For each factor:
(3)   LodPrevAtCurr(factor) = Move(LodPrev(factor), DA)
(4) SComm = SPrevAtCurr & SCurr
(5) RotMat = LodCurr(SComm) * pinv(LodPrevAtCurr(SComm))
```

The above procedure can be made separately for the dimensions of the image, in that the vertical loadings for the current temporal block depend only on the vertical loadings for the previous temporal block, and the horizontal loadings for the current temporal block depend only on the horizontal loadings for the previous temporal block:

```
LodVCurr = Move(RotMatV*LodVPrev)+ResVLod
LodHCurr = Move(RotMatH*LodHPrev)+ResHLod
```

The rotation matrices, RotMatV and RotMatH, can then be estimated separately for vertical and horizontal dimensions using the method described above.

Alternatively, it can be made jointly, using both vertical and horizontal loadings for the previous block for both vertical and horizontal loadings for the current block:

```
LodVCurr = Move(RotMatVV*LodVPrev+RotMatHV*LodHPrev)+ResVLod
LodHCurr = Move(RotMatVH*LodVPrev+RotMatHH*LodHPrev)+ResHLod
```

Or, equivalently written:

```
LodVCurr = Move(RotMatV*[LodVPrev; LodHPrev])+ResVLod
LodHCurr = Move(RotMatH*[LodVPrev; LodHPrev])+ResHLod
```

The rotation matrices, RotMatV and RotMatH, can again the estimated separately for vertical and horizontal dimensions using the method described above.

Related methods, although used in a different context and for a slightly different purpose, can be found in appendix MERGE_SUBSEQUENCES in the patent application "Method and Apparatus for Data Analysis", already included by reference.

In the above, the rotation matrix was computed by projecting the loadings of the current temporal block on the loadings of the previous temporal block. This is optimal with regard to sum of squares of the residual. However, it need not be optimal with regard to the compressibility of the residuals, especially if the residuals are compressed with a method exploiting spatial redundancy. Therefore, as a replacement for or as a postprocessing after projection, the compressibility of the residuals may be optimized in a separate step. One method is to use simplex optimizing. One program for performing simplex optimization is fmins, described in "Matlab Reference Guide" for Matlab version 4.2, 1992, pages 208-210, which is hereby included by reference. An object function to be optimized must first be defined. The object function can take as free variables a vector containing the elements of a rotation matrix. The object function then rotates the previous loading according to the given rotation matrix, computes residuals as the difference between the current loadings and the rotated loadings, and measures the compressability of the residuals, e.g. by compressing them and measuring the number of bits. The simplex program then proposes different rotation matrices to the object function, trying to optimize the compressability of the residuals.

Eighteenth preferred embodiment

In the previous preferred embodiment, the loadings for one temporal block were expressed in terms of loadings of the previous temporal block. This gives a possible compression advantage when the loadings of the blocks are similar.

In many cases, estimation of a motion field is an underdetermined problem. One case is the well known aperture problem, which appears in cases when little texture is present in the images. Another is for repeated textures. In such cases, where several motion fields are possible with similar reconstruction fidelity or quality, it would be advantagous to choose the motion field which results in the best compression rates for the overall system. One method which can be used within one temporal block is given in the patent application "Method and apparatus for coordination of motion determination over multiple frames", PCT application PCT/EP 96/01272, which is hereby included by reference.

Related methods can be used across temporal block limits.

After one temporal block has been compressed and transmitted, the loadings can be kept for the next temporal block. The new temporal block can then be modelled by projecting motion fields on the already existing loadings. Only the part of the motion fields that do not fit into the subspace spanned by the available loadings need be transmitted, preferably as new loadings.

This is illustrated in the following encoding algorithm, expressed as pseudo-code:

```
Initialize the previous loadings PrevLod to empty
For each temporal block:
  Select a reference frame
  Initialize CurrMod to empty
  For each other frame within the temporal block:
    Estimate motion from reference image to the current frame
    Project motion on PrevLod, producing PrevSco and
    motion residual
    Update CurrMod using motion residual

  Transmit PrevSco, and the score and loading of the
  first factor of CurrMod
  Include the loading of the first factor of CurrMod
  into PrevLod
```

This can be enhanced in various ways. One enhancement is to include a Move operation, so that the loads of the current temporal block are reconstructed as motion compensated loads from the previous temporal block. This can be done according to the following encoding algorithm:

```
Initialize the previous loadings PrevLod to empty
For each temporal block:
  Select a frame as reference image
  Initialize CurrMod to empty
```

```
For each other frame within the temporal block:
  Estimate motion from reference image to the current frame
  Project motion on PrevLod, producing PrevSco and
  motion residual
  Update CurrMod using motion residual

Compute a motion field DA from the previous
reference image to the current reference image

Transmit DA
For each loading in PrevMod:
  Move according to DA, producing a loading of CurrMod
Transmit PrevSco, and the score and loading of the
first factor of CurrMod
Include the loading of the first factor of CurrMod
into PrevLod
```

Another is to limit the number of factors. According to the above methods, there will be one new factor for each new temporal block. This may not be acceptable. One solution is to periodically initialize PrevLod to empty. These points may be used as random access points.

Another solution is to check the previous loadings for relevance, e.g. by checking how much of the variance of the motion fields of the current temporal block can be expressed by projection on the different loadings of the previous temporal blocks, and deleting those factors that have the least ability to explain variance.

Yet another solution is to periodically, when the number of factors is approaching maximum, to estimate the most relevant subspace of the loadings, perform a factor rotation, and skip the least relevant factors. The estimation is done on the encoder side, a rotation matrix is transmitted, and the rotation is performed both on the encoder and decoder side.

The same principles can be applied for bi-linear modelling of intensity changes.
The following pseudo-code illustrates one way of combining the above enhancements:

```
Initialize the previous loadings PrevSmiLod and PrevBluLod
to empty
For each temporal block:
  Select a frame as reference image
  If there was a previous temporal block:
    Compute a motion field from the previous reference
    image to the current reference image, either by
    multiplying scores with loadings if
    the current reference image was covered by the
    previous temporal block, otherwise by motion estimation.
  Motion compensate PrevSmiLod and PrevBluLod using
  the computed motion

  Initialize CurrSmiMod and CurrBluMod to empty
```

```
For each other frame within the temporal block:
  Estimate motion from reference image to the current frame
  Project motion on PrevSmiLod, producing PrevSmiSco +
  motion residual

  Update CurrMod using motion residual

Transmit PrevSco, and the score and loading of the
first factor of CurrMod
Include the loading of the first factor of CurrMod
into PrevLod
```

Nineteenth preferred embodiment

In some applications, not the entire content of each frame should be modeled. One such application is content based editing of movies, e.g. when an actor is encoded separately from his background, so that she can later be superimposed on another given background.

In such a case, one image area can be selected manually or automatically in a reference image, and this image area is then followed throughout a sequence.

Using manual or automatic segmentation methods, a region of interest can be selected in a reference image. One method is chroma keying, often also called blue screening. Another method is manual segmentation. Another group of methods is segmenting based on intensity or motion. One such method is described in "Method and apparatus for multi-frame based segmentation of data streams", PCT patent application PCT/EP 96/01273, which is hereby included by reference.

One possible representation of such a region of interest is to use an image of same size as the reference image, with ones for those pixels that are to be included and zeros for the others. Such a representation is sometimes called an alpha channel of an image. In PCT/EP 96/01273, it is called an S field.

The patent application "Method and apparatus for data analysis", already included by reference, describes how to build an IDLE model when an S field is given.

The patent application "Apparatus and method for decoding video images", already included by reference, describes how to decode an IDLE model when an S field is given.

When an S field for the reference image is given, and motion is estimated for the temporal block, then a reconstructed S field for each frame can be made when reconstructing the frames.

Additionally, an S field for each frame may be given as input to the encoder. In the case that the reconstructed S field does not coincide with the input S field for each frame, i.e. the shape of the reconstructed image area is not as wanted, then an S residual can be transmitted. The S residual

can be formed by an artihmetic difference between original and reconstruction, in which case it will in general be a tri-level image.

Alternatively, it can be formed using an exclusive-or operation between original and reconstruction, in which case it will be a bi-level image. All compression techniques that have been listed for compression of B fields can be used for compression of S fields or S residuals.

When several image areas have been modeled throughout one or more sequences, they can be mixed together. One procedure for doing this is given in the patent application Method and Apparatus for Decoding Video Images, already included by reference. In that patent application, the modeled image areas are called holons.

Twentieth preferred embodiment

The method of the present invention can be used in applications like interactive video or video games.

One method is to adapt the decoding order of frames based on user input. One simple implementation is to decode selected temporal blocks depending on user input. A small example is given:

```
printf("Please insert credit card")
Decode("MovingCreditCardSequence")
AwaitCreditCard
printf("Please select service")
input(Service)
If Service==Cash
     Decode("PINCodeGuidance")
     ...
elseif Service = AccountStatement
     Decode("AnimatedPrintout")
     ...
end
```

The example shows a fraction of a script controlling an automatic teller machine equipped with an image display. Various instructive sequences are decoded, based on user input. Each sequence can consist of one or more temporal block of frames.

Another method is to adapt the content of decoding based on user input. Any part of an IDLE model may be changed or controlled to produce a wanted effect. One example is shown on fig. 25, showing a decoder 2500 for interactive use. A storage device, exemplified by a CD-ROM 20, stores rules describing the action in the game together with IDLE models. The rules are input to an ActionController 2530. A first IDLE model is stored in a model store ModSto 40. An input device 2510 is controlled by the user. This could be a joystick, mouse or similar. The input is fed to an action controller 2530. The action controller 2530 controls the various actions in the

game by producing scores that are used for producing reconstructed frames. One example is animating a given user controlled character. Input values can be directly connected to score values. When the user moves his joystick or mouse to the right, then a score for a horizontal factor is increased, causing the user controlled character to move to the right. Input can also be indirectly connected to score values: When the user controlled character moves to the right, then scores for another horizontal factor are activated in a cyclic fashion, giving the impression of walking legs. Score values can also depend on other events: After a given period of time, or when the user controlled character has performed some specific action, then a given sequence of score values is activated. The scores and the IDLE model in ModSto 2540 are used to reconstruct frames in Rec 2550, and these are displayed on a Display 2560. The action controller may also initiate loading of a new IDLE model from the storage medium 2520.

Score values may be connected to other data sources. E.g, if both sound samples and a corresponding bi-linear model of motion are available for a talking person, then the scores for the motion may be synchronized with the replay of the sound, so that the lips move according to the sound.

For games, it may be particularily useful to use S fields, as described in the previous preferred embodiment. S fields can be used to move foreground characters independently of other characters or background.

While the invention has been particularily shown and described with reference to the preferred embodiments thereof, it will be understood by those skilled in the art that various changes in form and detail may be made therein without departing from the spirit and scope of the invention.

CLAIMS

1. A method for encoding a sequence of video frames,
each frame comprising pixels,
wherein
the sequence is divided into temporal blocks, each
temporal block comprising a plurality of video frames,
each temporal block is represented as one
reference frame and a bilinear model of motion, the
reference frame and bilinear model of motion together
comprising an IDLE model,
the bilinear model of motion comprising a plurality
of column vectors called score vectors and a plurality of
row vectors called loading vectors, so that each element
of each score vector corresponds to one of the frames and
so that each element of each loading vector corresponds
to one dimension of the motion for a pixel for the
Reference Frame,
the combination of one score vector and one loading vector
together being called a factor,
the motion for each video frame comprising the sum of factor
contributions from all factors,
and each factor contribution being the loading for the factor
scaled with the score element corresponding to the frame
for the factor,
wherein the collection of Reference Frames and the
collection of factors together represent the sequence
in encoded form.

2. The method according to claim 1,
the method comprising the steps of:
(1) dividing the sequence into a plurality of
temporal blocks, each temporal block comprising a
given number of frames, each frame comprising pixels,
(2) for each temporal block, performing steps (2) to (8),
(3) selecting one reference frame called I frame,
(4) for each other frame, called U frame, in the temporal
block, estimating motion from the I frame to the U frame,
thereby yielding a motion field,
(5) for each motion field produced in step (4),
reformatting the motion field into a row vector,
assembling the row vectors into a motion matrix,
(6) based on the motion matrix producing in step (5),
computing a bilinear model of motion,
(7) selecting a predefined number of factors, and
(8) transmitting or storing the I frame and the scores
and loadings comprising the selected number of factors,
wherein the results transmitted or stored in step (8)
together comprise the sequence in compressed representation,

so that a reconstruction of the sequence of video frames can be computed from the compressed representation.

3. The method according to claim 2, wherein the bilinear model is updated successively as each motion is estimated.

4. The method according to any one of claims 1 to 3, wherein a first reference frame is used as a basis for reconstructing a second reference frame, the second reference frame being called a P frame.

5. The method according to claim 4, wherein a first and a second temporal block overlap such that a P frame of the second temporal block can be reconstructed using an IDLE model corresponding to the first temporal block.

6. The method according to any one of claims 1 to 5, wherein selection of reference frames is adaptive.

7. The method according to claim 6, wherein the choice of I frames is made so that there is maximally a given number of frames between two successive I frames.

8. The method according to any one of claims 2 to 7, wherein a first bilinear model of motion is used to form a prediction of loadings of a second bilinear model of motion,
and the loadings of the second bilinear model are represented as a corrector being the difference between the predicted loadings and the loadings of the second bilinear model of motion.

9. The method according to claim 8, wherein the forming of the prediction of the loadings of the second bilinear model of motion comprises the steps of
(6 b) moving the loadings corresponding to one temporal block of frames into a position corresponding to the reference frame of the other temporal block of frames, thereby producing moved loadings, and
(6 c) projecting the moved loadings on the loadings corresponding to the other temporal block of frames, thereby producing a rotation matrix,
wherein the rotation matrix represents the prediction.

10. The method according to any one of claims 1 to 9, wherein a plurality frames, called B frames, are selected, and each B frame is represented according to two reference frames and their associated bilinear models of motion.

11. The method according to claim 10,
wherein for each B frame, a B field is given, the B field indicating for each pixel, block of pixels or group of pixels in the B frame the ratio of the reference images to be used in the reconstruction of the B frame.

12. The method according to claim 11,
wherein the B fields are simplified according to a compromise between simplicity and reconstruction fidelity.

13. The method according to any one of claims 1 to 12,
the method also comprising spatially extending the IDLE model so that each reconstructed frame will be fully covered when the reference frame is moved according to the bilinear model of motion.

14. The method according to claim 13,
wherein the extending of the IDLE model comprises the steps of:
(7 b) increasing the size of the reference frame, thereby obtaining a reference image,
(7 c) increasing the size of the loadings,
(7 d) extrapolating the loadings spatially,
(7 e) for one or more of the frames, performing step (7 f) to (7 h),
(7 f) reconstructing the frame according to the reference image and the bilinear model of motion,
(7 g) moving the part of the corresponding original frame not covered by the reconstruction back to the coordinate system of the reference image using the inverse of the motion for the frame, and
(7 h) including the parts moved back in step (7) as new parts of the reference image, thereby obtaining an extended reference image.

15. The method according to any one of claims 1 to 14,
wherein the selection of frames to be used as P frames is made dependent on one of or a combination of the following criteria:
(1) minimum transmission cost,
(2) fidelity of frames reconstructed according to either estimated motion or a bilinear model of motion,
(3) how well a motion field for a new frame fits to a bilinear model of motion computed for earlier frames, and
(4) a given limit on the allowed number of frames between any two consequtive I frames.

16. The method according to claim 15,
wherein the fidelity in step (2) is computed by dividing the residual image into blocks, and counting how many blocks there are with an average error larger than a given threshold.

17. The method according to any one of claims 1 to 16, wherein the number of factors for a bilinear model of motion is chosen according to one of or a combination of the following criteria:
(1) number of frames in the bilinear model of motion,
(2) energy content in factors,
(3) relative reconstruction fidelity or predicted subjective quality of frames reconstructed with varying number of factors present, and
(4) available transmission capacity.

18. The method according to any one of claims 1 to 17, the IDLE model also comprising a bilinear model of intensity changes,
the bilinear model of intensity changes comprising a plurality of column vectors called score vectors and a plurality of row vectors called loading vectors, so that each element of each score vector corresponds to one of the frames and so that each element of each loading vector corresponds to one pixel of the reference frame,
the combination of one score vector and one load vector together being called a factor,
the intensity changes for each video frame comprising the sum of factor contributions from all factors,
and each factor contribution being the loading for the factor scaled with the score element corresponding to the frame for the factor.

19. The method according to claim 18, wherein the bilinear model of intensity changes is computed by the following steps:
(1) for each current frame in the temporal block, performing steps (2) to (4),
(2) reconstructing the current frame using the IDLE model of the temporal block,
(3) subtracting the original frame from the reconstruction, producing a difference image,
(4) moving the difference using the inverse of the motion used in step (2), and
(5) computing a bilinear model of the results produced in step (4).

20. The method according to claim 18, wherein the bilinear model of intensity changes is computed by the following steps:
(1) for each current frame in the temporal block, performing steps (2) to (4),
(2) computing a motion field corresponding to the current frame by multiplying the score vectors corresponding to

the current frame by the loading vectors,
(3) moving the current frame according to the inverse of the motion field produced in step (2),
(4) subtracting the reference frame from the result produced in step (3), producing a difference image, and
(5) computing a bilinear model of the results produced in step (4).

21. The method according to any one of claims 1 to 17, the IDLE model also comprising a bilinear model of intensity changes,
the bilinear model of intensity changes comprising a plurality of column vectors called score vectors and a plurality of row vectors called loading vectors, so that each element of each score vector corresponds to one of the frames and so that each element of each loading vector corresponds to one pixel position in a frame,
the combination of one score vector and one load vector together being called a factor,
the intensity changes for each video frame comprising the sum of factor contributions from all factors,
each factor contribution being the loading for the factor scaled with the score element corresponding to the frame for the factor,
the bilinear model of intensity changes being computed according to the following steps:
(1) for each current frame in the temporal block, performing steps (2) to (3),
(2) reconstructing the current frame using the reference frame and corresponding bi-linear model of motion,
(3) subtracting the original frame from the reconstruction, producing a difference image, and
(4) computing a bilinear model of the results obtained in step (3).

22. The method according to any one of claims 19 to 21, wherein the computing of a bilinear model of the difference images utilizes weights based on the visual significance of the difference image.

23. The method of any one of claims 1 to 22, wherein one or more of the I frame, score and loading is compressed before being transmitted or stored.

24. The method according to claim 23, wherein at least one loading is compressed according to a transform into frequency.

25. The method according to claim 24,

wherein the transform into frequency comprises a Discrete Cosine Transform, DCT.

26. The method according to any one of claims 23 to 25, wherein the compression quality for each loading depends on the scores of the same factor, higher absolute values of the scores resulting in higher fidelity for compression of the loading.

27. The method according to any one of claims 4 to 26, wherein for each P, B or U frame, the B, B or U frame is reconstructed, a residual is computed as a difference between reconstructed and original P, B or U frame, and the residual is compressed before being transmitted or stored.

28. The method according to claim 27, wherein characteristics of the frames or the bilinear model of motion according to the human visual system are exploited when compressing residuals for P, B or U frames, in order to achieve a good compromise between amount of data to be transmitted and subjective quality of the reconstructed frames.

29. The method according to any one of claims 1 to 28, wherein the video sequence is represented in color, and one or more of the reference image, bilinear intensity model or residual are represented in color accordingly.

30. The method according to claim 18 and 29, wherein the bilinear model of intensity changes is adapted to color, so that each each element of each loading of the bilinear model of intensity changes corresponds to one color component of one pixel of a reference image.

31. The method according claim 18 and 29, wherein one bilinear model of intensity is produced for each color channel.

32. The method according to any one of claim 29 to 31, wherein one or more color component of one or more of the reference image, bilinear model of intensity changes or residual is represented in reduced spatial resolution.

33. The method according to any one of claims 1 to 32, wherein the video frames are represented in interlaced format, each frame comprising an even and an odd video field, the reference image is formed from two video fields, taking into consideration the motion between the video fields, and the bilinear model of motion is formed so as to compensate for the temporal shift between video fields of a reconstructed

video frame.

34. The method according to claim 33,
wherein the fields of the video sequence are treated as
if they were frames according to any one of claim 1 to 32.

35. The method according to claim 33,
wherein even video fields are treated as one video sequence
and odd video fields are treated as another video sequence
according to any one of claim 1 to 32.

36. The method according to claim 33,
wherein the reference frame is formed as a composite of
two fields, such that one field is motion compensated to
fit to the other.

37. The method according to any one of claim 23 to 28,
wherein the intended or available bit rate for the compressed
representation is used to influence one of or a combination
of the following:
(1) number of chosen factors,
(2) selection of I and P frames, and
(3) compression fidelity for reference images, loadings or
scores.

38. The method according to any of claim 1 to 37,
wherein for each reference frame, a field,
called S field, is given, defining for each pixel in the reference
frame whether the pixel should be modelled.

39. The method according to claim 38,
wherein for each frame to be reconstructed, the S field is moved
according to the bi-linear model
of motion, and the moved S field is used to determine the visibility
of zero or more corresponding reconstructed pixels.

40. The method according to claim 39,
wherein for each frame, an input S field is given, indicating for each
pixel of the frame whether the corresponding reconstructed pixel should
be decoded, and this field is transmitted and compressed using the
moved S field as a predictor.

41. The method according to any of claim 38 to 40,
wherein one or more of the reference image or the bilinear model of
motion is compressed utilizing the S field to achieve higher
compression ratios.

42. The method according to any of claim 38 to 41,
wherein a collection of a reference frame, a corresponding
bilinear model of motion and an S field together comprise

a holon, and
a reconstruction is made as a synthesis from two or more holons.

43. The method according to any one of claims 1 to 42, wherein a bilinear model of motion for one temporal block is used to influence choice of representation of the following temporal block.

44. The method according to any one of claims 1 to 43, wherein the motion fields or bilinear model of motion may be represented in subsampled resolution.

45. The method according to any one of claims 1 to 44, wherein one bilinear model is used for each of the vertical and horizontal dimension of the motion respectively.

46. The method according to any one of claims 1 to 45, adapted to be used for interactive operation.

47. A method for decoding a sequence of video frames, each frame comprising pixels,
characterized in that
the sequence is divided into temporal blocks, each temporal block comprising a plurality of video frames, each temporal block is represented as one Reference Frame and a bilinear model of motion, the bilinear model of motion comprising a plurality of column vectors called score vectors and a plurality of row vectors called loading vectors, so that each element of each score vector corresponds to one of the frames and so that each element of each loading vector corresponds to one dimension of the motion for a pixel for the Reference Frame,
the combination of one score vector and one loading vector together being called a factor,
the motion for each video frame comprising the sum of factor contributions from all factors,
and each factor contribution being the loading for the factor scaled with the score element corresponding to the frame for the factor.

48. The method according to claim 47, adapted to be used together with the method of any one of claims 2 to 46.

49. The method according to any one of claims 47 to 48, adapted to the method according to any one of claims 10 to 12 such that a B frame is reconstructed from only one reference frame and its bilinear model of motion in

case the other reference frame or bilinear model of motion is not available.

50. An apparatus for encoding a sequence of video frames, the apparatus comprising
means for dividing the sequence into temporal block, each temporal block comprising a plurality of video frames,
means for selecting a Reference Frame,
means for representing each temporal block as one Reference Frame and a bilinear model of motion,
the bilinear model of motion comprising a plurality of column vectors called score vectors and a plurality of row vectors called loading vectors, so that each element of each score vector corresponds to one of the frames and so that each element of each loading vector corresponds to one dimension of the motion for a pixel for the Reference Frame,
the combination of one score vector and one loading vector together being called a factor,
the motion for each video frame comprising the sum of factor contributions from all factors,
and each factor contribution being the loading for the factor scaled with the score element corresponding to the frame for the factor,
wherein the Reference Frame and the bilinear model of motion together comprise the encoded sequence of video frames.

51. The apparatus according to claim 50,
adapted according to the method of any one of claim 2 to 49.

52. A data structure representing a sequence of video frames,
characterized in that
the sequence is divided into temporal blocks, each temporal block comprising a plurality of video frames,
each temporal block is represented as one Reference Frame and a bilinear model of motion,
the bilinear model of motion comprising a plurality of column vectors called score vectors and a plurality of row vectors called loading vectors, so that each element of each score vector corresponds to one of the frames and so that each element of each loading vector corresponds to one dimension of the motion for a pixel for the Reference Frame,
the combination of one score vector and one loading vector together being called a factor,
the motion for each video frame comprising the sum of factor contributions from all factors,
and each factor contribution being the loading for the factor

scaled with the score element corresponding to the frame for the factor.

53. The data structure according to claim 52, adapted according to claim 2 to 51.

Fig. 1

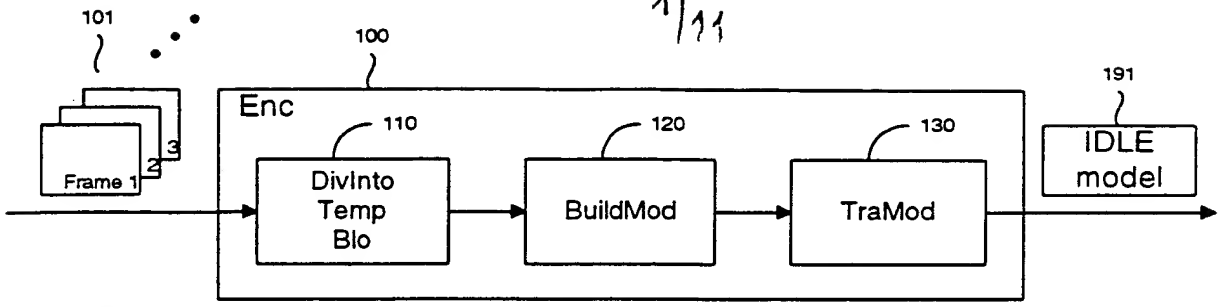

Fig. 2

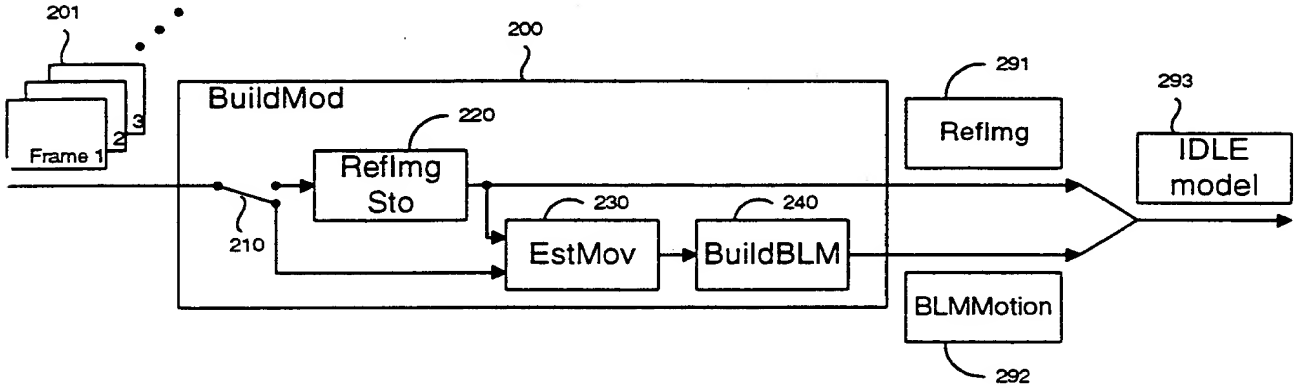

Fig. 3 a

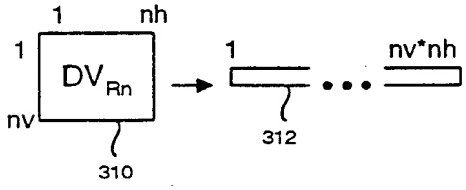

Fig. 3 b

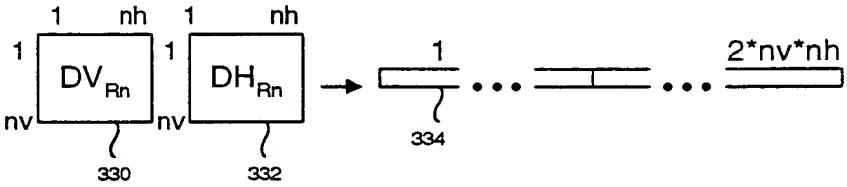

Fig. 3 c

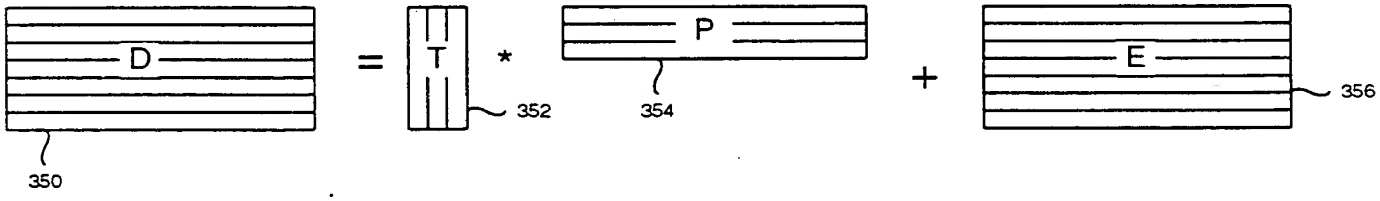

Fig. 3 d

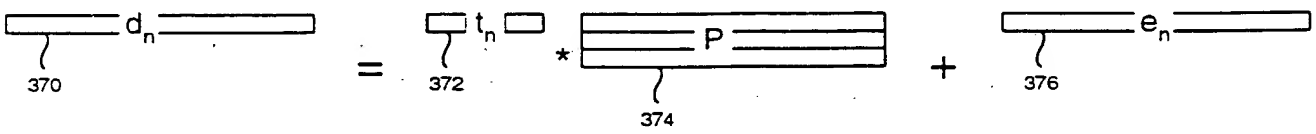

Fig. 4

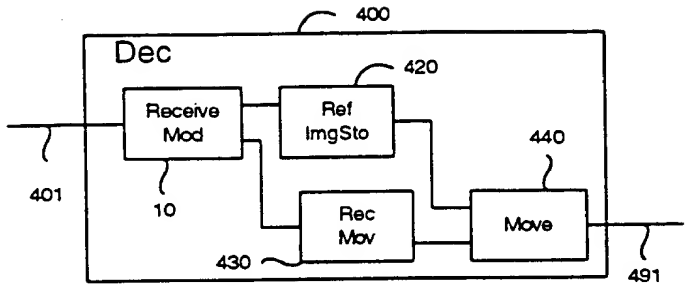

Fig. 5

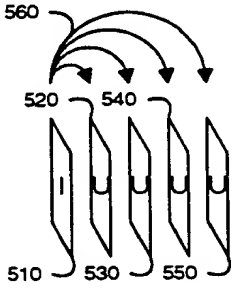

Fig. 6

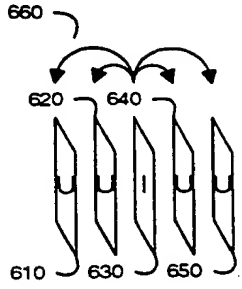

Fig. 7

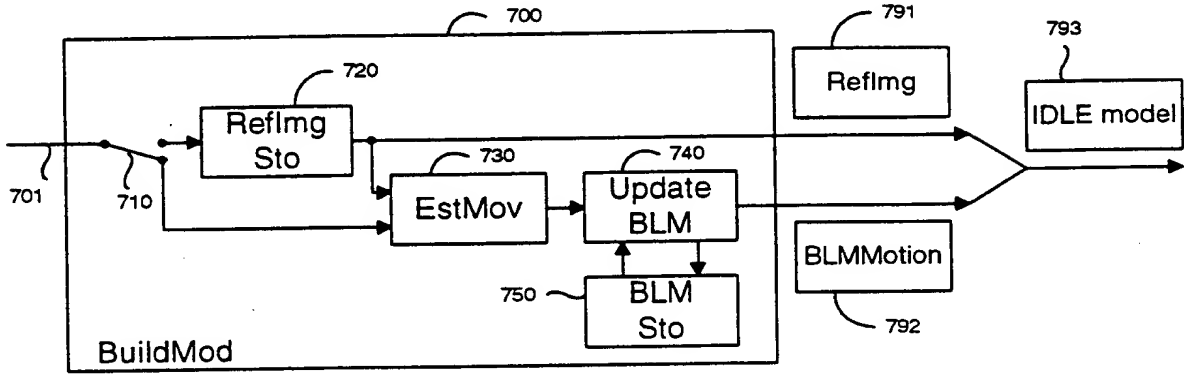

Fig. 8 a

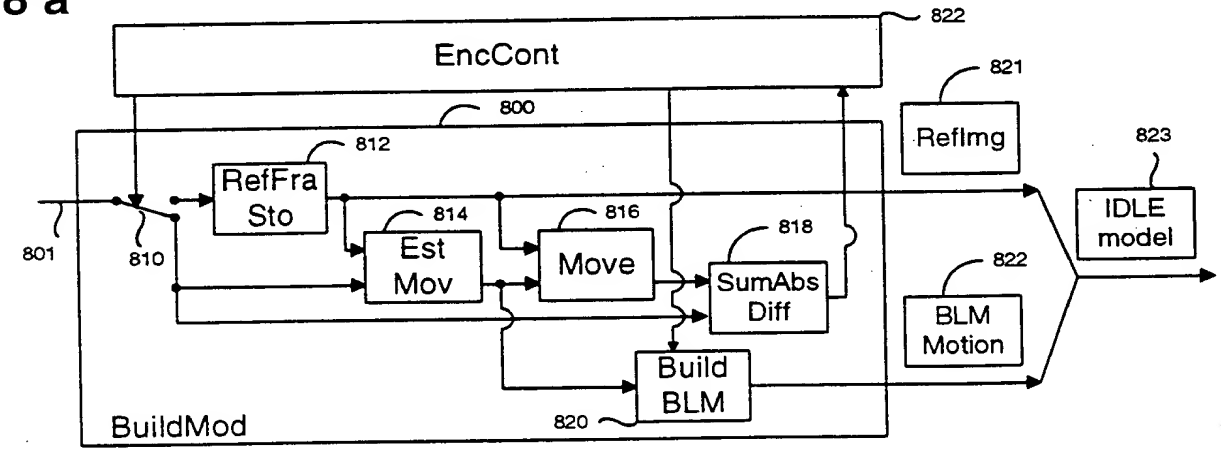

Fig. 8 b

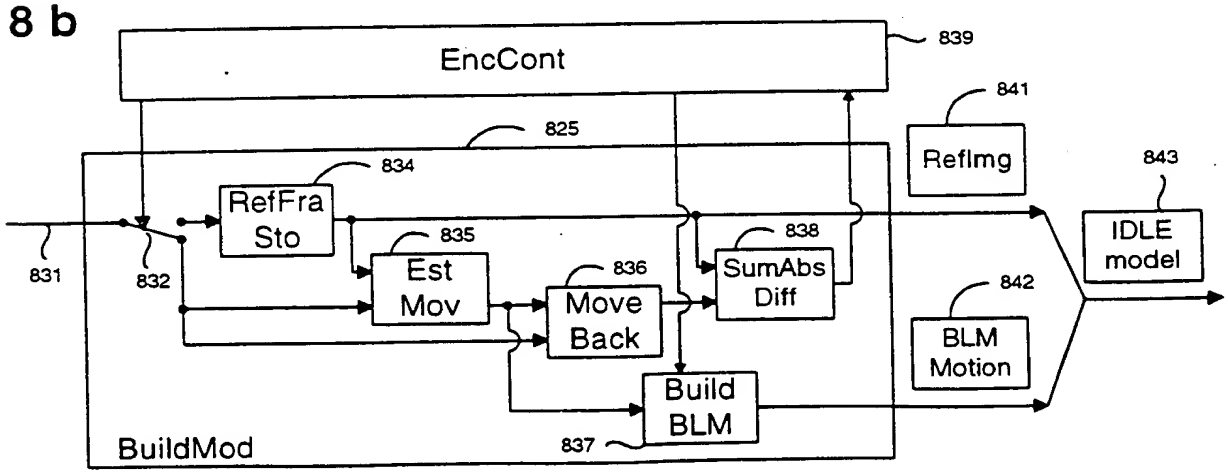

Fig. 8 c

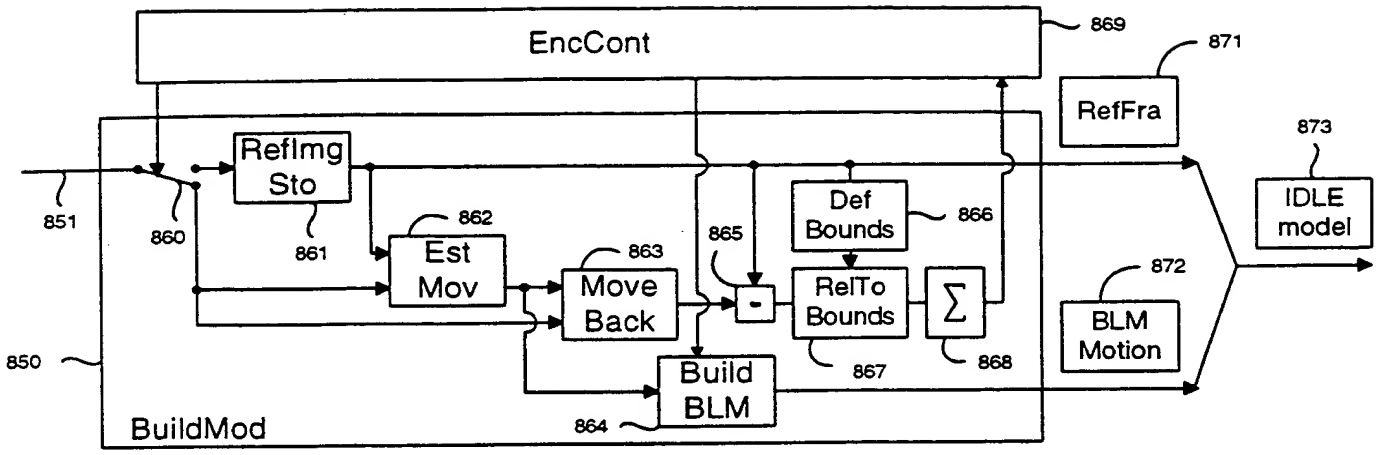

Fig. 8 d

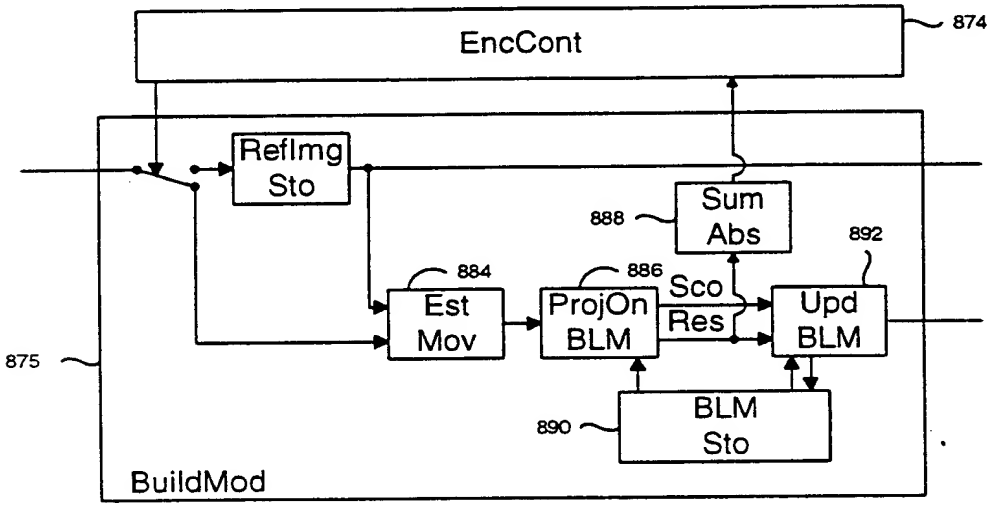

Fig. 9

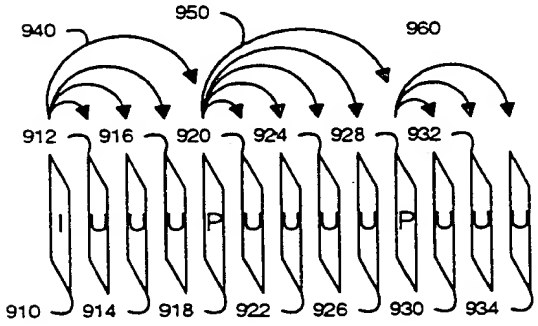

Fig. 10

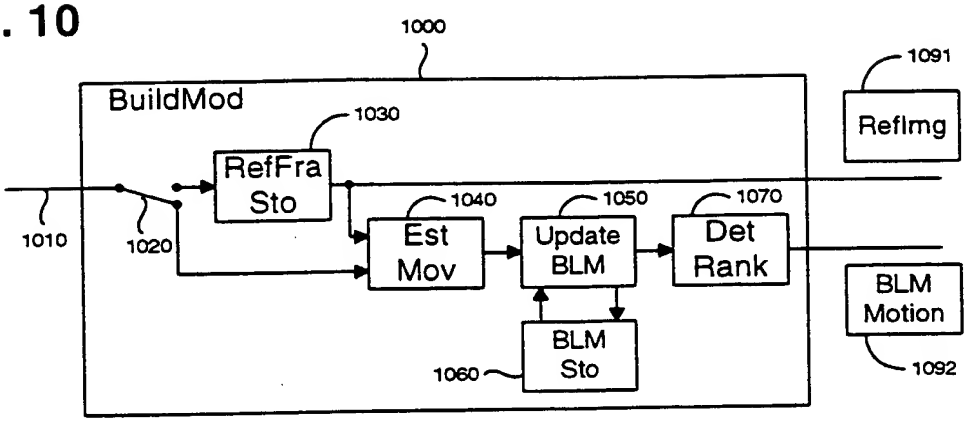

Fig. 11 a

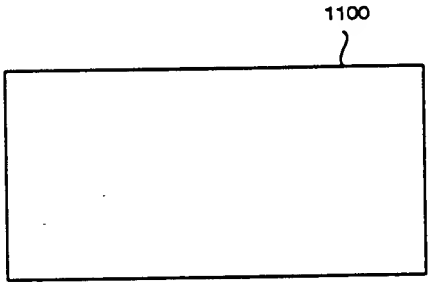

Fig. 11 b

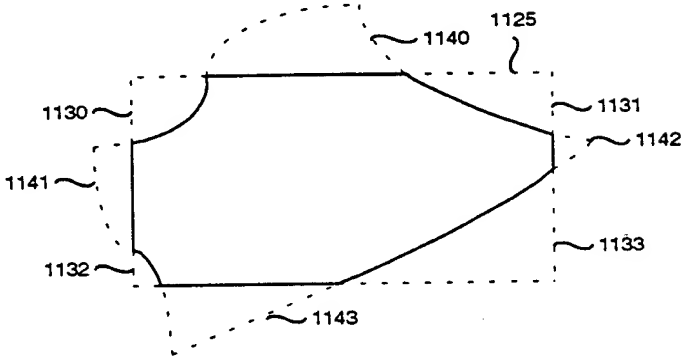

Fig. 11 c

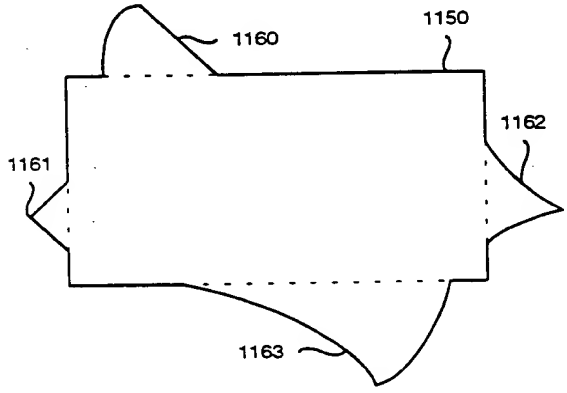

Fig. 11 d

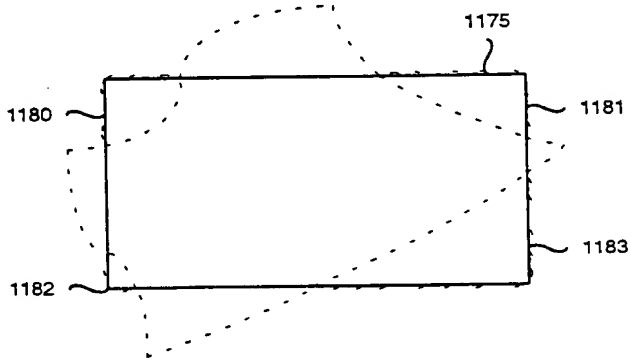

Fig. 12 a

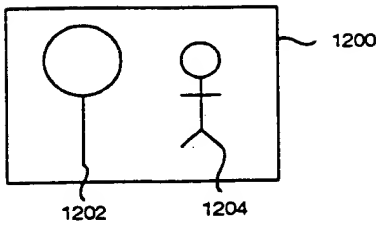

Fig. 12 b

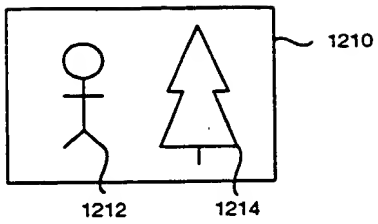

Fig. 12 c

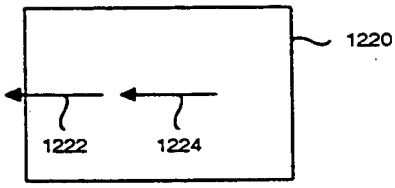

Fig. 12 d

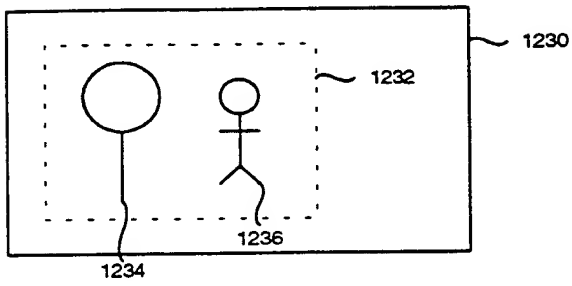

Fig. 12 e

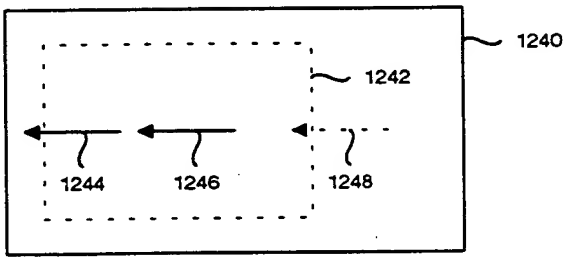

Fig. 12 f

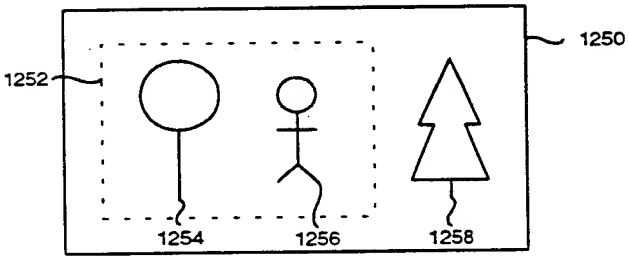

Fig. 12 g

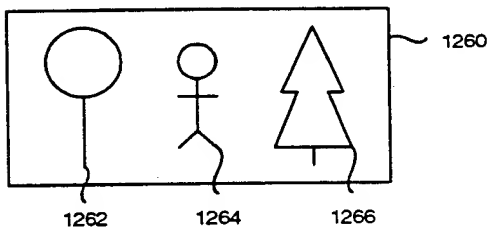

Fig. 12 h

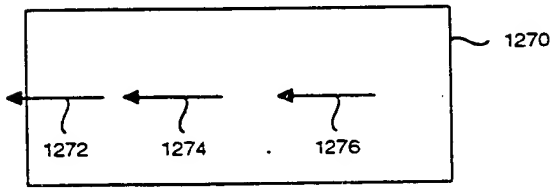

Fig. 13 a

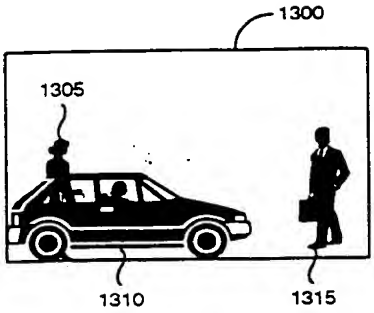

Fig. 13 b

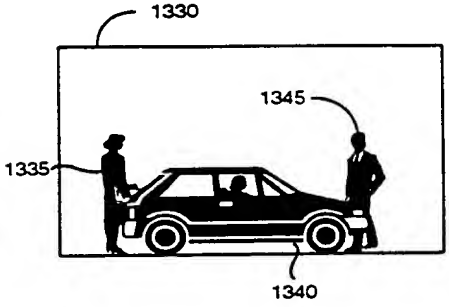

Fig. 13 c

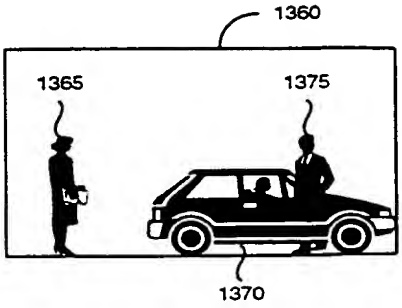

Fig. 14

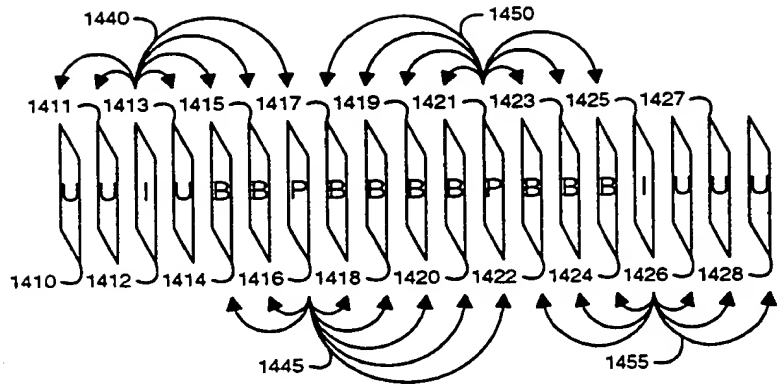

Fig. 15

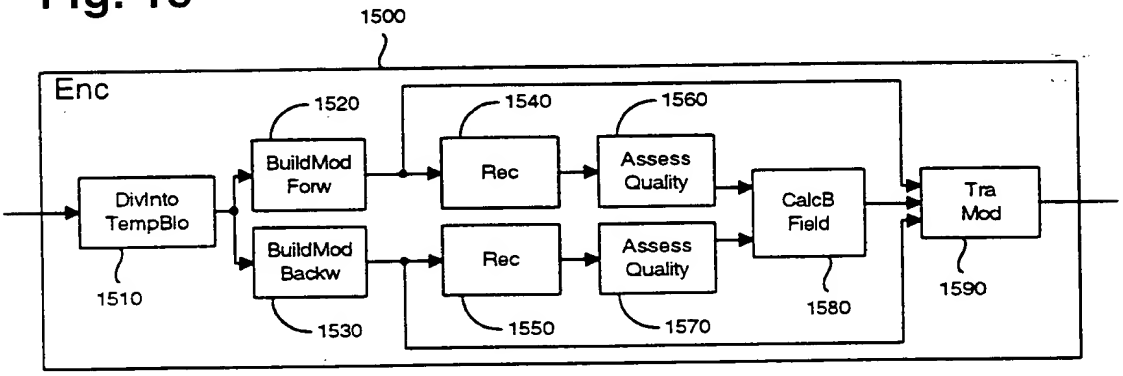

Fig. 16

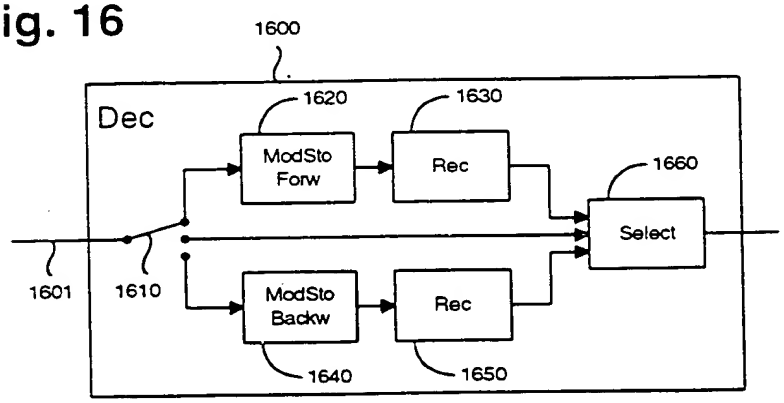

Fig. 17 a

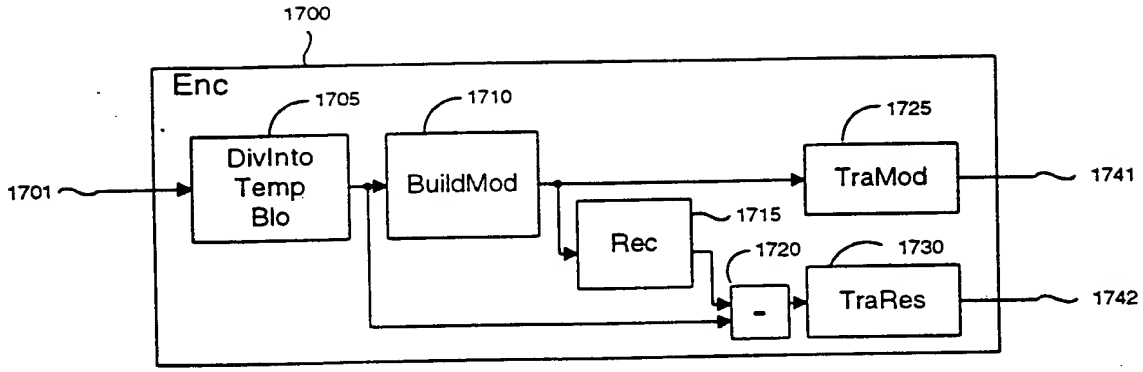

Fig. 17 b

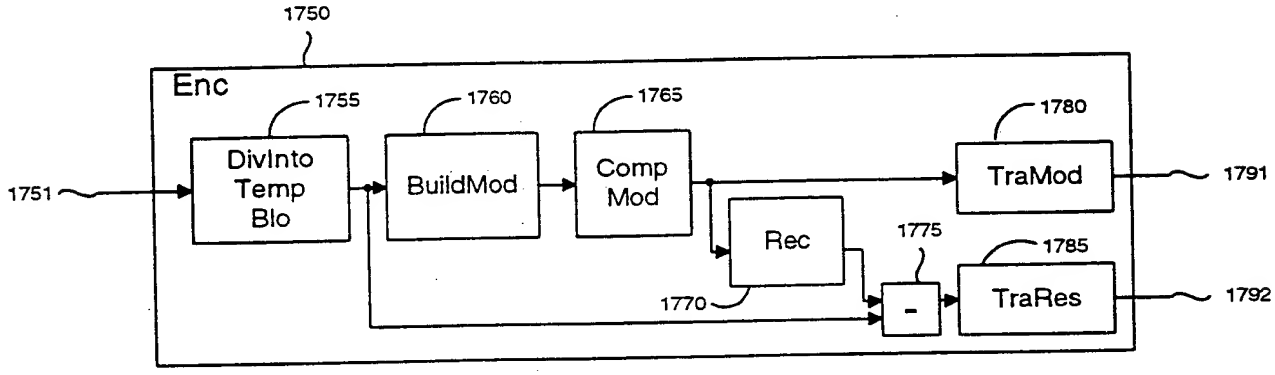

Fig. 18

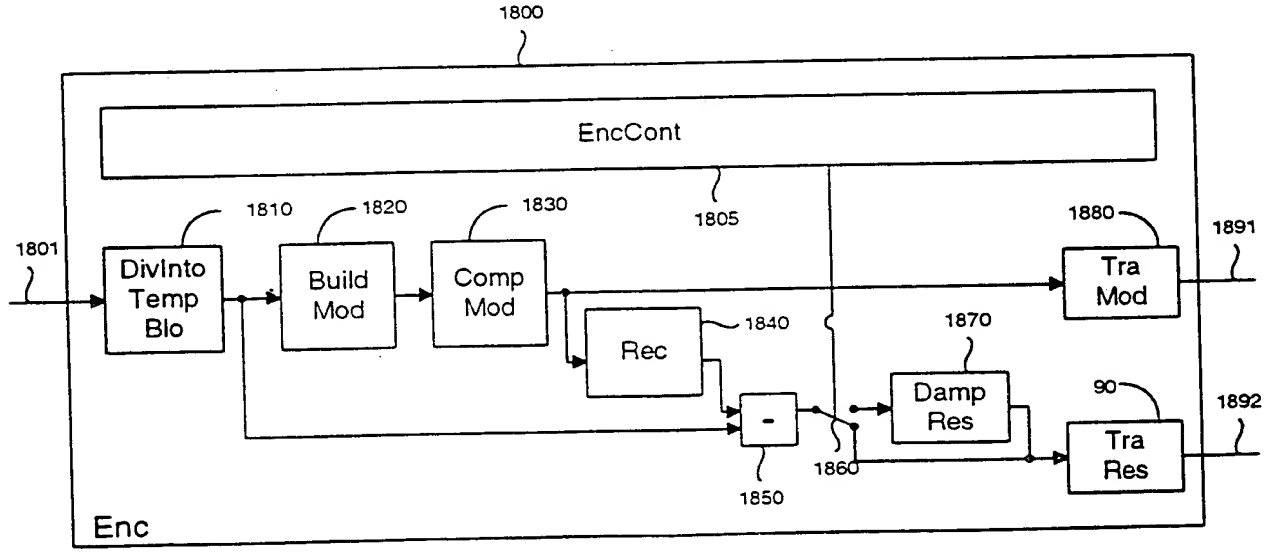

Fig. 19 a

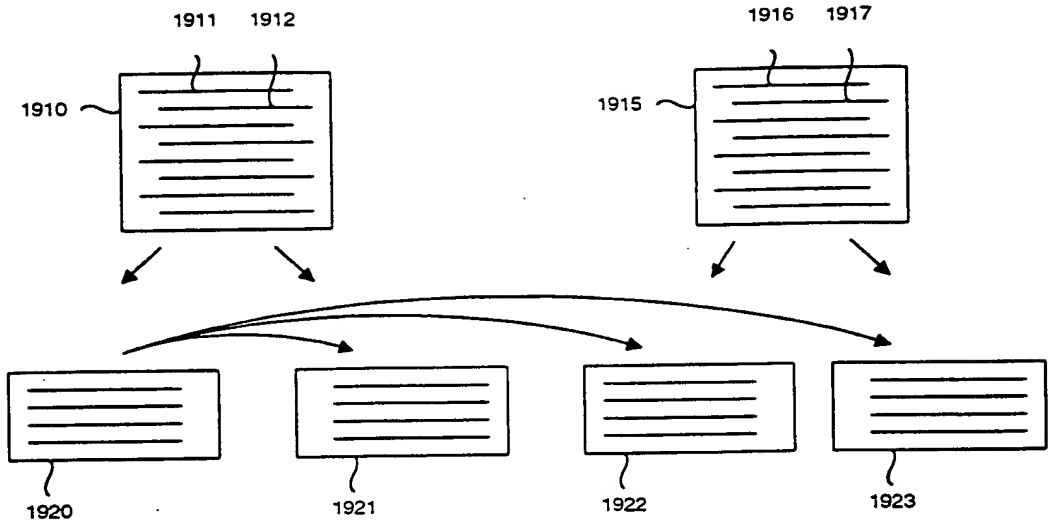

Fig. 19 b

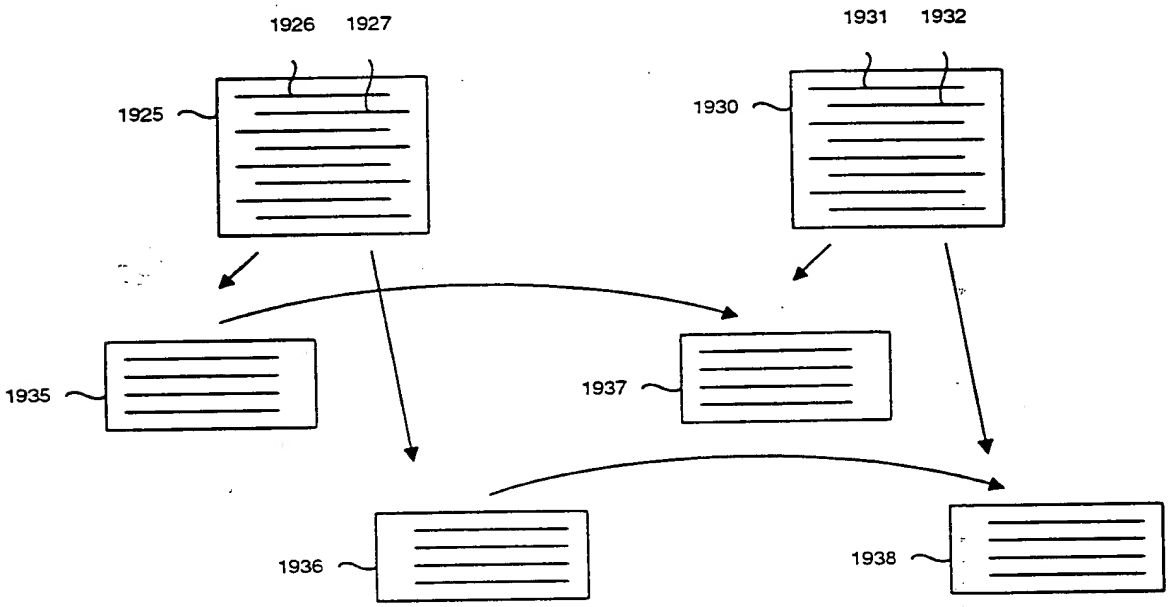

Fig. 19c

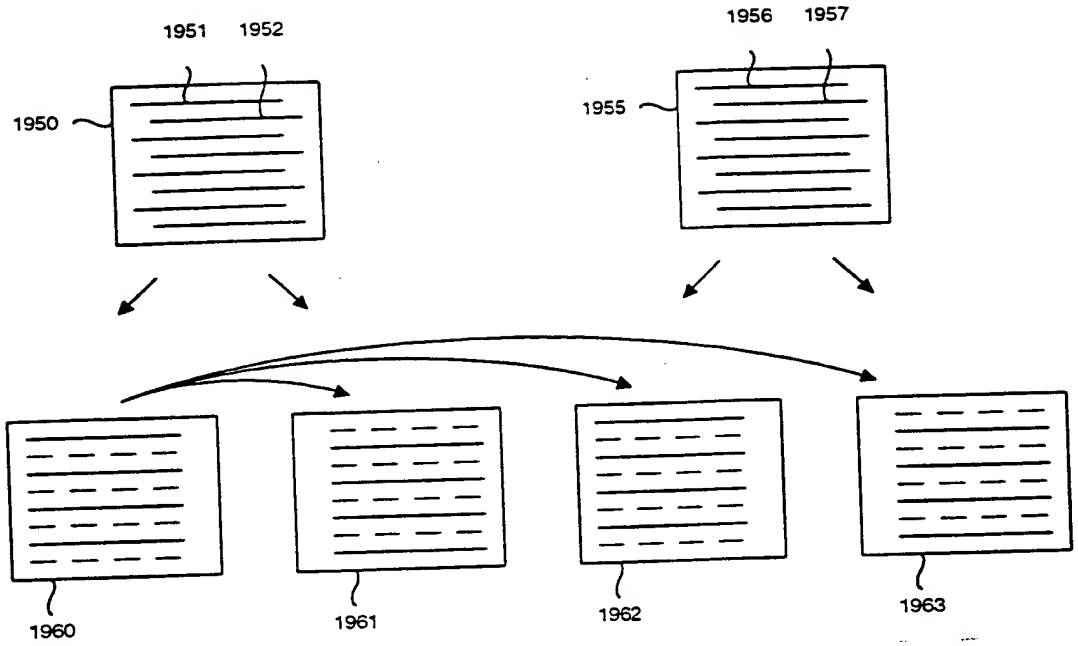

Fig. 19 d

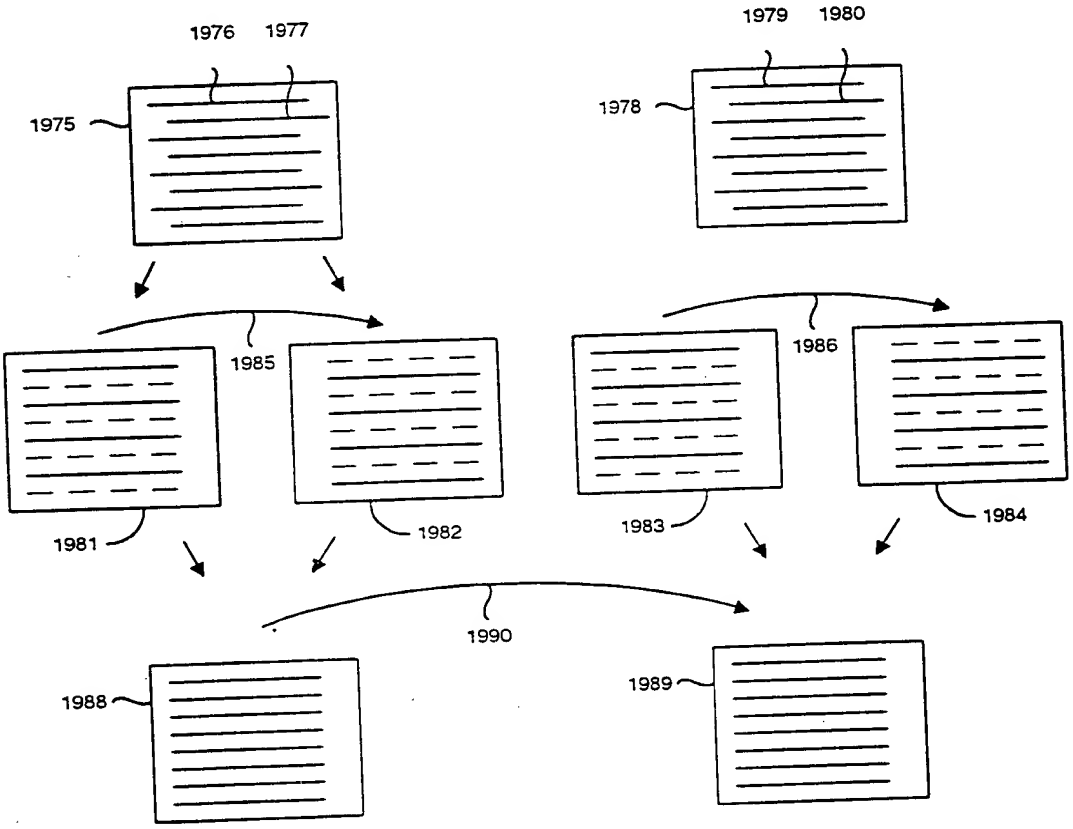

Fig. 20

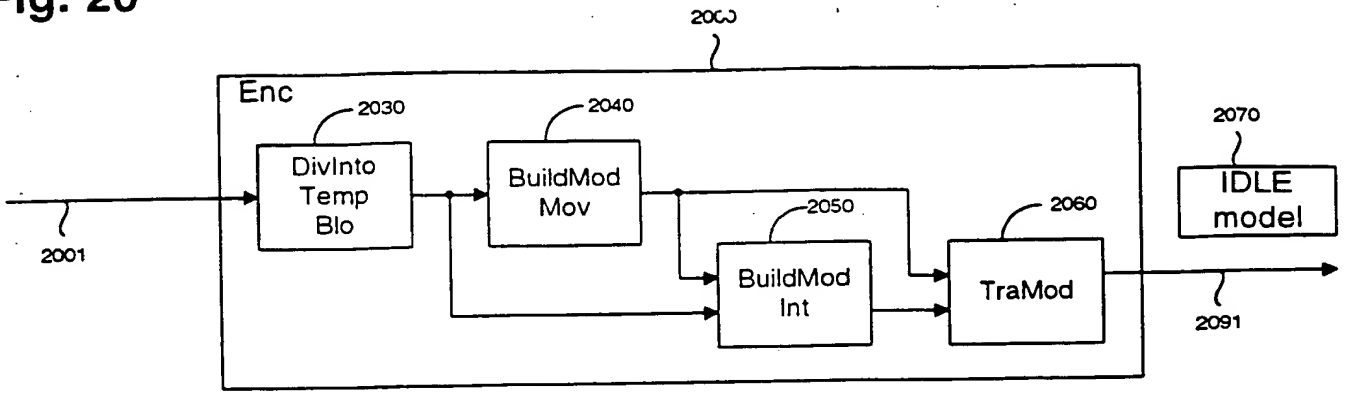

Fig. 21 a

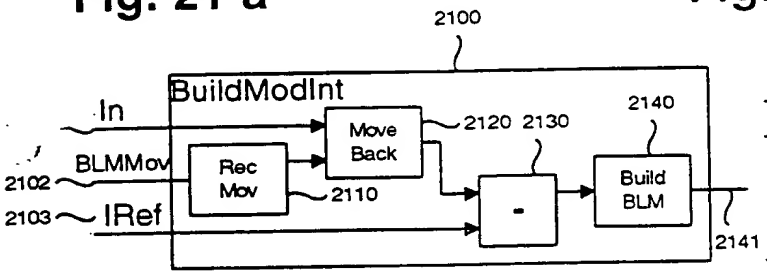

Fig. 21 b

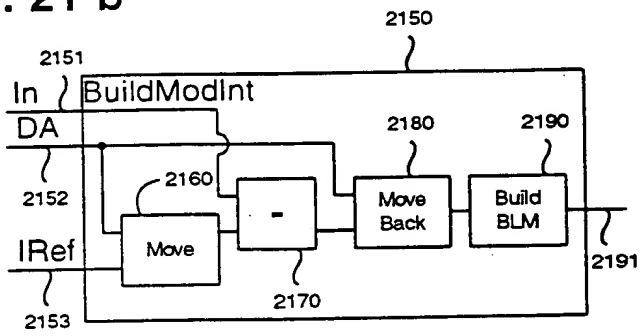

Fig. 22

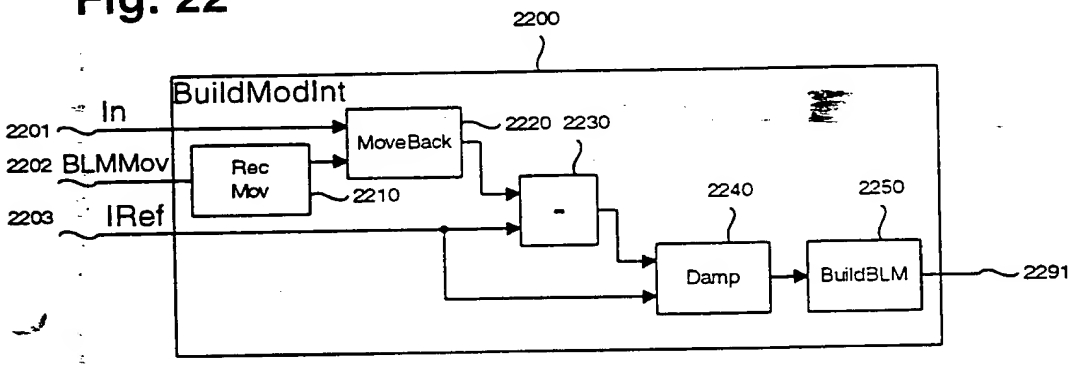

Fig. 23 a

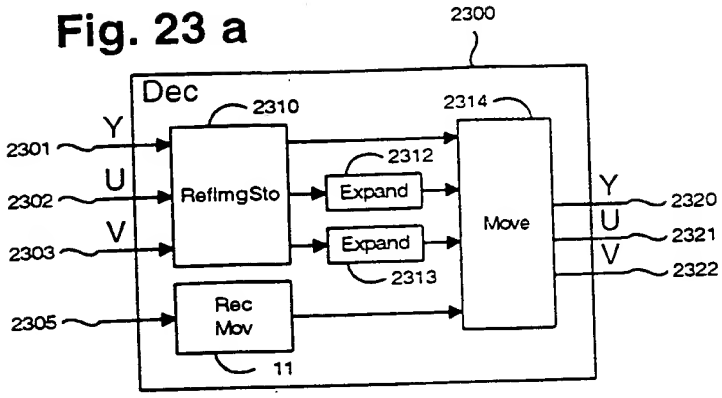

Fig. 23 b

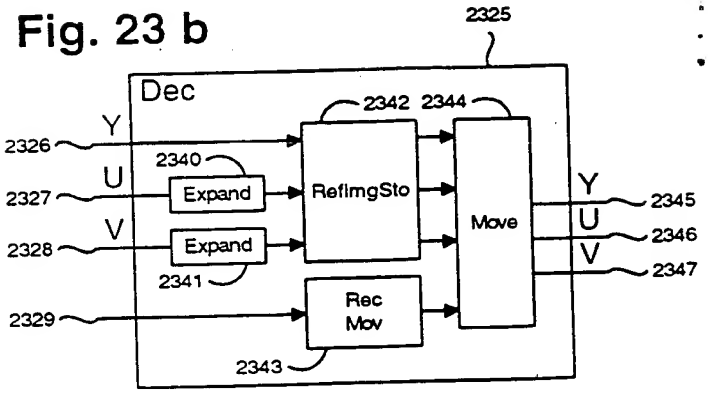

Fig. 23 c

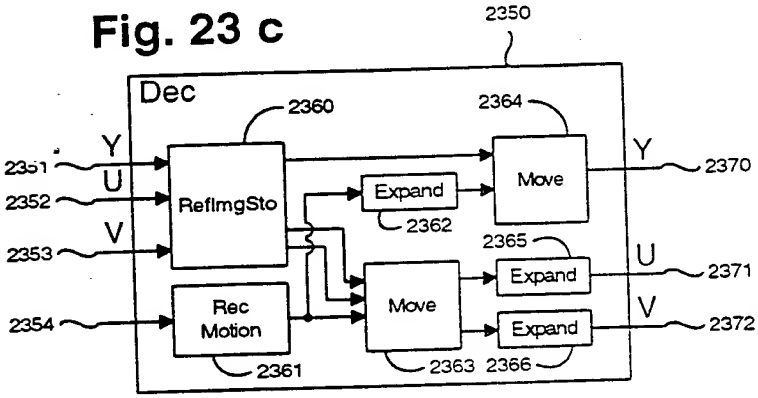

Fig. 23 d

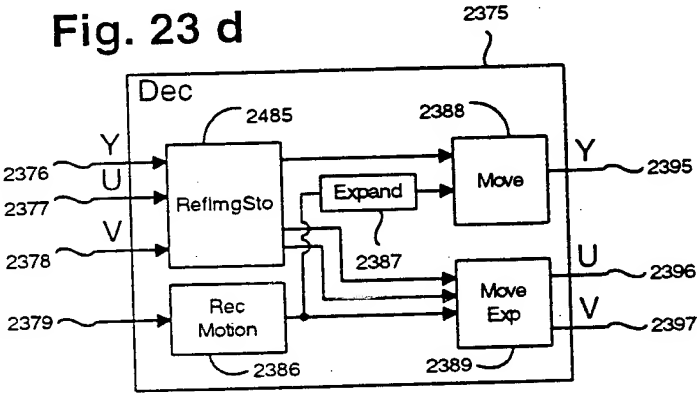

Fig. 24

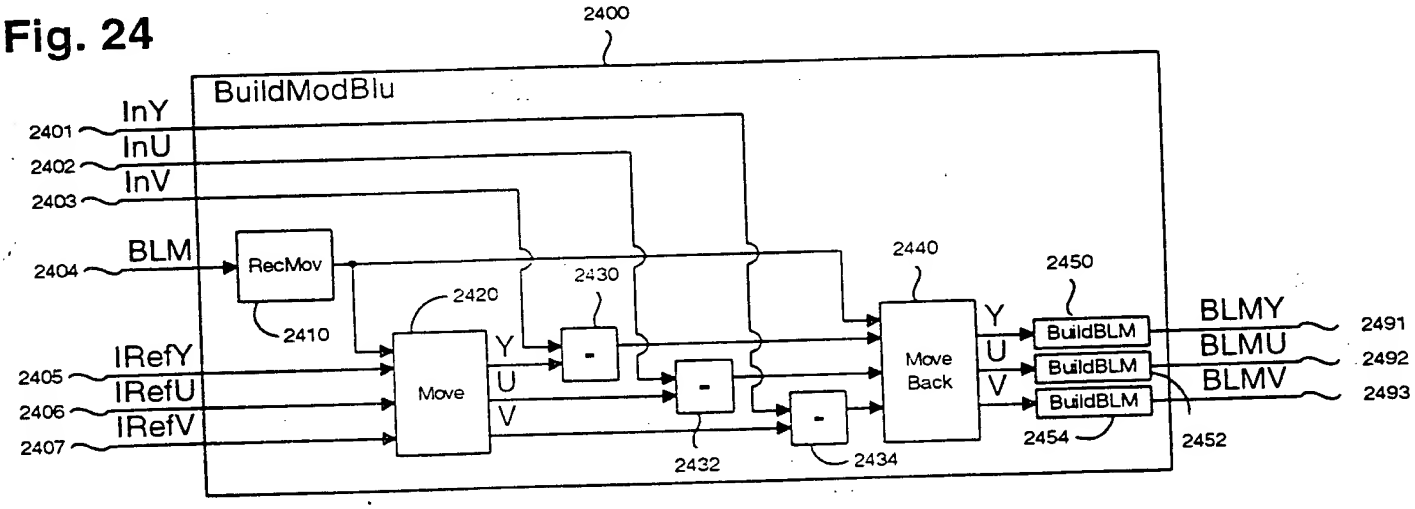

Fig. 25